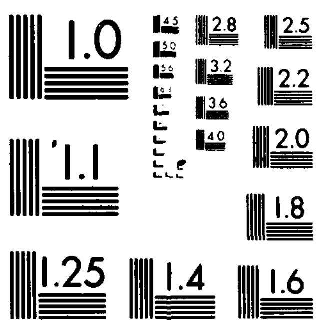
AD-A115 399 NORTHEASTERN UNIV BOSTON MASS ELECTRONICS RESEARCH LAB F/6 7/4
CONTROL ELECTRONICS FOR AIR-BORNE QUADRUPOLE ION MASS SPECTROME--ETC(U)
OCT 81 J S ROCHEFORT, R SUKYS F19628-78-C-0218

UNCLASSIFIED AFGL-TR-82-0056 NL

1 of 4


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0

(12)

AFGL-TR-82-0056

AD A115399

CONTROL ELECTRONICS FOR AIR-BORNE QUADRUPOLE ION MASS SPECTROMETER

J. Spencer Rochefort
Raimundas Sukys

Northeastern University
Electronics Research Laboratory
Boston, Massachusetts 02115

Approved for public release; distribution unlimited.

October 1981

Final Report

Term Covered: 15 September 1978 through 14 September 1981

AIR FORCE GEOPHYSICS LABORATORY
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
HANSCOM AFB, MASSACHUSETTS 01731

82 06 10 045

DISCLAIMER NOTICE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

MIL-STD-847A
31 January 1973

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFGL-TR-82-0056	2. GOVT ACCESSION NO. AD A115 899	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Control Electronics for Air-Borne Quadrupole Ion Mass Spectrometer		5. TYPE OF REPORT & PERIOD COVERED Final. 15 Sept. 1978 14 Sept. 1981
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) J. Spencer Rochefort, Raimundas Sukys		8. CONTRACT OR GRANT NUMBER(s) F19628-78-C-0218
9. PERFORMING ORGANIZATION NAME AND ADDRESS Northeastern University, Electronics Research Laboratory, Boston, MA 02115		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61102F 2310G3AM
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Geophysics Laboratory, Hanscom AFB, Massachusetts 01731, Monitor/L.E. Wrodyka/LKD		12. REPORT DATE October 1981
		13. NUMBER OF PAGES 308
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release; Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Programmable Control
Microprocessor
Mass Spectrometer Control

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The work concerned in this report spans the three-year period commencing 15 September 1978 and deals with the development of electronic instrumentation for rocket and balloon-borne quadrupole ion mass spectrometers. Electronic packages containing circuits to control the quadrupole filters and data processing were developed for sounding rockets launched from

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Red Lake, Ontario, Canada during the 26 February 1979 Solar Eclipse; from the Poker Flat Research Range, Chatanika, Alaska during 1980 and 1981 Solar Proton Event Programs and also the 1981 Auroral E Program. A microprocessor-based control system was developed for a balloon-borne mass spectrometer. This latter system also provides a two-way communications link for ground control of the experiment and data transmission during the flight.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

TABLE OF ILLUSTRATIONS

TABLE OF SELECTED ACRONYMS AND ABBREVIATIONS

1. AMU. . . . . ATOMIC MASS UNIT
2. BBIMS. . . . . . . BALLOON BORNE ION MASS SPECTROMETER
3. BΔ . . . . 204.8kHz CLOCK SIGNAL
4. BINFO RST. . . . . SIGNAL TO CLEAR THE MASS SPECTROMETER DATA COUNTER
5. BQ . . . . MASS SPECTROMETER DATA TO THE COMBINATIONAL LOGIC SYSTEM
6. CD VALID . . . . . SIGNAL INDICATING THAT A VALID COMMAND HAS BEEN RECEIVED THROUGH THE TONE COMMAND LINK
7. CIMS . . . . CLUSTER ION MASS SPECTROMETER
8. CL . . . . COMBINATIONAL LOGIC
9. CLK1 . . . . MASS SPECTROMETER DATA INPUT TO A COUNTER
10. CLK2 . . . . CLOCK SIGNAL FOR ELAPSED FLIGHT TIME COUNTER
11. CPU/COMB LOGIC . . CENTRAL PROCESSING UNIT OR COMBINATIONAL LOGIC CONTROL
12. CSX AND ASX. . . . CONTROL SIGNALS TO THE ANALOG MULTIPLEXER
13. D/A. . . . . DIGITAL TO ANALOG
14. D TO A . . . . . . DIGITAL TO ANALOG
15. DATA READY . . . . SIGNAL THAT DATA IS READY FOR TRANSMISSION
16. DATA VALID . . . . A FLAG INDICATING THAT NEW DATA IS PRESENT IN THE PCM FRAME
17. DBIN . . . . INPUT DATA BUS TO MEMORY
18. DBOUT. . . . . . . INPUT DATA BUS TO MEMORY
19. DBX. . . . . ONE OF EIGHT DATA LINES TO THE PARALLEL TO SERIAL CONVERTER IN THE PCM SYSTEM
20. FCU. . . . . FLIGHT CONTROL UNIT
21. 1ST. FRAME . . . . SIGNAL TO LOAD SYSTEM STATUS INTO PCM BUFFER
22. FS . . . . SIGNAL INDICATING THAT THE FIRST BYTE OF THE PCM FRAME SYNC IS READY FOR TRANSMISSION

TABLE OF SELECTED ACRONYMS AND ABBREVIATIONS (continued)

23. GCU. . . . . GROUND CONTROL UNIT

24. GO . . . . SIGNAL INDICATING THE START OF DATA COLLECTION PERIOD

25. INC DOM. . . . . . INCREMENT AMU SELECTION SIGNAL BY ONE STEP

26. MS . . . . MASS SPECTROMETER

27. NO DOWN. . . . . . DATA CORRECTION FOR NOISE INDUCED ERRORS NOT NECESSARY

28. PFRR . . . . *POKER FLAT RESEARCH RANGE*

29. P/S. . . . . SIGNAL TO LOAD PARALLEL TO SERIAL DATA CONVERTER

30. Q. . . . . COMMON BIAS OF THE QUADRUPOLE

31. Qx . . . . TRANSISTOR; FLIP-FLOP OR COUNTER OUTPUT DESIGNATION

32. RAM DUMP . . . . . TRANSMISSION OF DATA FROM RANDOM ACCESS MEMORY

33. S-TO-P . . . . . . SERIAL TO PARALLEL

34. SELX . . . . MASS SPECTROMETER CONTROL SIGNALS ORIGINATING IN THE μP BASED CONTROL SYSTEM

35. SPE. . . . . SOLAR PROTON EVENT

36. SRX. . . . . SIGNALS TO THE USERS OF THE DIGITAL DATA BUS IN THE PCM SYSTEM

37. UART . . . . UNIVERSAL ASYCHRONOUS RECEIVER TRANSMITTER

38. UP/DOWN . . . . . CONTROL SIGNAL FOR THE DATA COUNTER TO COUNT UP OR DOWN

39. USART or USRT. . . UNIVERSAL SYNCHRONOUS/ASYNCHRONOUS RECEIVER/ TRANSMITTER

40. WSMR . . . . WHITE SANDS MISSILE RANGE

INTRODUCTION

This final report is primarily concerned with the development of electronic instrumentation firmware and software associated with balloon-borne quadrupole ion mass spectrometers. Work associated with rocket-borne instrumentation is summarized. Some of the work reported originated under the prior contract[1]. The remainder of the work discussed can be classified as the refurbishment of recovered instruments or the construction of somewhat modified versions of previously designed packages. The material which follows is grouped into two chapters.

The first chapter is concerned with rocket programs. The instrumentation provided was an outgrowth of that developed under the previous contract. System concepts have remained the same and the majority of the modifications from one vehicle to the next have been made to accommodate variations in spectrometer design or the scientific objectives of the experiment itself. Consequently the changes incorporated into this generation of instruments have usually been those required to realize different voltage ranges, quadrupole exciter frequencies, data word length, packaging constraints and modernization of existing circuits. This being the case, the majority of the discussions found in Chapter 1 are quite brief. Detailed discussion of the electronic systems associated with the rocket-borne instruments may be found in References 1 and 3.

The second chapter is concerned with a balloon-borne system. This work started under the previous contract and was continuously carried on until the completed system was delivered to the Air Force in August 1981.

During the contract two Scientific Reports[2,5] dealing with the balloon-borne systems were issued. An in-depth discussion of the instrumentation developed is carried in this chapter. Previously issued in-house type publications concerned with flight and ground control routines are included in the appendices.

I. ROCKET PROGRAMS

A. THE 1979 SOLAR ECLIPSE

Two cluster ion mass spectrometer (CIMS) electronic packages were completed for the AFGL Solar Eclipse Program. Work on these units was initiated during the prior contract. Anticipating less sophisticated ground support equipment at the Red Lake Launch Site, Ontario, Canada, than had been previously available at WSMR, 10-bit synchronization and data words were employed in the PCM data transmission system. One rocket package was completely constructed while the second resulted from the modification of a package which had not been launched from WSMR earlier. In the latter case, new flight control units had to be constructed. A new design and package layout for the high-voltage section was incorporated in these vehicles. The quadrupole excitation signal and control circuits were substantially the same as in the WSMR rounds. Descriptions of the control circuits may be found in Reference 1. The ac exciter circuits, developed in part as a sponsored Master of Science Thesis by T. Palasek, were described in the Scientific Report No. 2 issued under this contract[3]. Field support was supplied for both vehicles during the period 8 February - 1 March 1979 at the Chukuni Launch Range, Ontario, Canada. Both packages were successfully flown on 26 February 1979. The first vehicle was launched during the totality and the second was launched approximately 45 minutes later.

B. THE SOLAR PROTON EVENT PROGRAMS

The CIMS instruments constructed under this program were essentially the same as those employed in the Solar Eclipse Program and thus the com-

mon background of development is once more to be found in the publications of the previous contract.

1. The 1980 SPE

Two CIMS instruments were prepared for an October 1980 launch at Poker Flat Rocket Range, Chatanika, Alaska. One instrument became available due to a cancellation of a launch at WSMR and consequently merely required modification. The other was completely constructed. With the exception of the bias section, the same type of the electronic subsystems used in the Solar Eclipse Program were also employed in the SPE instrumentation. In the bias section, the cumbersome current sources and voltage dropping devices used to generate the accelerator and common-rod bias signals were replaced by a high-voltage operational amplifiers. The target high-voltage supply (Venus K30-Z) was moved from the RF section of the instrument onto the same deck with the multiplier high-voltage supply. The latter was instrumented with a F-50 model to avoid the increasing costs of the previously used MG12. The RF oscillator frequency was reduced to 1.85MHz thus allowing operation up to 255 amu without pushing the oscillator output capabilities to the limit.

The two instruments were delivered to AFGL and one was subsequently launched on 22 October 1980 and then recovered. Personnel from this contract were not involved in the field party.

2. The 1981 SPE

The recovered instrument from the 22 October 1980 launch was refurbished and checked out along with the vehicle which had not been launched. Payload and integration tests were conducted at AFGL. Contract personnel provided field support for these two vehicles at PFRR, Alaska, during the

period 5-19 August 1981. Both vehicles were installed on launchers and a solar proton event, SPE, awaited. Since a SPE did not appear likely in late August the field personnel returned on 19 August 1981 and went into a standby mode.

The field party returned to Alaska under the follow-on contract on 13 October 1981. One vehicle was launched on 26 October 1981.

C. THE AURORAL E. PROGRAM

A previously flown (1975) switched positive ion/neutral instrument was refurbished for the Auroral-E program conducted during March of 1981 at PFRR, Alaska. The instrument predated the CIMS and had sustained some damage during the previous flights. Therefore, some of the damaged and outdated circuits were improved and simplified. The broken transformer core of the RF oscillator was replaced by a core of the same vintage. But the RF amplitude control circuits which preceeded the comparator peak detector were modified. The flight programmer was completely redesigned and constructed in order to meet the new program requirements set by the scientist. An interface and safety interlock unit was also incorporated in the refurbished package.

The new programmer design was built about the 2758 EPROM. Two-byte control words were used to define the bias voltages and the mass filter excitation signals. A breakpoint was generated whenever a digital comparator detected a match between a counter output and those digits in the control word which were used to define a specific amu number. All together two separate programs containing 256 breakpoints each could be run. The vehicle timer selected the programs.

The interface unit was designed to interconnect the mass spectrometer, the telemetry system, the power transfer circuits, the flight timer and the umbilical connections. It replaced a previously used unit to provide additional space for other experiments in the vehicle. The relay used to switch between internal and external power sources was included in this unit. Buffer amplifiers were used to isolate the monitor signals going into the telemetry unit from the loading and noise pickup created when the umbilical cable was connected. An interlock arrangement in the timer unit was provided to prevent the possibility of simultaneous conflicting commands issued by the vehicle timer and the ground controller.

The electronics package was delivered to AFGL in October 1980 and subsequently followed through payload integration and testing. Contract personnel were not involved in the field program, but the rocket was successfully launched on 6 March 1981.

II. BALLOON BORNE ION MASS SPECTROMETER

A. OVERVIEW OF THE BBIMS PROGRAM

The development of the highly flexible electronics system to control a balloon borne ion mass spectrometer (BBIMS) had its origins under the previous contract. Some of the basic functions which were needed were incorporated in a prototype microprocessor-based design developed under a sponsored Electrical Engineer Thesis by V. Gerousis. This work was published under this contract as Scientific Report[2] No. 1.

A detailed description of the final system developed under this contract was presented as a conference paper[4] and subsequently issued as Scientific Report[5] No. 3. Since that publication carries a detailed discussion of the system and its capabilities the remaining sections of this chapter will be devoted to descriptions of the major circuits comprising the system together with their circuit diagrams. Previously issued in-house publications concerned with flight and ground control routines are included in the Appendices.

Functionally the electronics system of the BBIMS was subdivided into three major subsystems: the exciter circuits, the control circuits, and the communication circuits. The primary function of the exciter circuits was to generate and to provide analog signals to the structure of the mass filter. The control circuits determined the parameters that placed the mass filter in a desired mode of operations within the mass spectrum. Data transmission and ground based commands to the airborne unit were in the realm of the communication circuits, which included a Ground Control Unit (GCU).

The digital control signals to the analog circuits passed through a D to A interface. From these signals the dc and the ac components of the quadrupole excitation signal were generated. The ratio between the two components was determined by a multiplying DAC. Another set of five DAC's were used to provide bias voltages to the filter and to the ion optics.

These basic tasks, with very limited communications and data handling capabilities, were incorporated in a preliminary microprocessor based design reported in Reference 2. Changing and expanding requirements for flexibility in the control and communications capability of the balloon borne instrumentation package required a redesign of the system. Noise induced error compensation, selected bias voltage sweep, controlled amu band fast spectrum scan, cummulative count and other modes of operation were incorporated. Combinations of these modes were also possible. Communications through a serial UP/DOWN links and/or through the serial UP and through the PCM DOWN links were introduced. Data gathering and transmission were synchronized with the PCM encoder. This new primary microprocessor based system was augmented with an independant secondary combinational/sequential CMOS logic system. The control of the exciter circuits was exercised through a common interface with the CPU.

The data from the balloon borne instruments were processed by a programmable CMOS PCM encoder. The received PCM data were demultiplexed and displayed for monitoring purposes by a GCU. Control over the airborne instrument was also exercised through GCU. The radio links were not included in the development and were furnished by AFGL.

B. THE EXCITER CIRCUITS

1. Power Transfer and Supplies

The power transfer circuits included in the mass spectrometer package are shown in Figure 1. The main power from the 28 volt balloon battery assigned to the spectrometer was controlled by a master relay not shown in the drawing. The same relay also transferred the power from an 8 volt battery to the μP based mass spectrometer control circuits. The master relay in turn was controlled through a tone command link. The power to the high voltage supplies and the RF oscillator were also controlled through the tone command system. Two channels of the link provided ground closures for the 2N2907 transistors to turn the power ON and OFF. Separate controls for the HV and the RF circuits were provided for convenience during laboratory operations.

The power supply circuits are shown in Figure 2. The battery voltage, preregulated to 20 volts, powered the dc-dc converter. The nonsaturating squarewave ac to ac converter used two FERROXCUBE 2616-3C8 pot core transformers to provide the necessary outputs for the bridge rectifier circuits HEXFET's driven by a 25kHz symmetrical squarewave derived from a CMOS oscillator provided the chopped dc to the transformers.

Three additional voltages were derived without the benefit of the transformer. The +20V was taken from the previously mentioned preregulator, +40 volts was obtained from the drive circuits of the HEXFET's (J,H) and the -20 volts was generated through a dc restorer circuit. The output circuits of the last two signals are shown in Figure 1. The -20V were used to switch the HV and the RF power relays OFF during the flight. The +40 volts were provided for busing. Voltage regulation was provided at the circuit level when required.

The high voltages for the positive ion target and the electron multiplier were derived from commercially available supplies.

2. Bias Circuits

The bias and the bias monitor circuits are shown in Figure 1. The digital control word was latched into the μP compatable DAC configured for bipolar operation. The output was amplified by a high voltage amplifier to the required level. Precision rectifiers provided unipolar monitor signal for transmission through the telemetry. Five such bias circuits were used in the BBIMS unit.

3. Vacuum and H.V. Monitors

The quadrupole housing vacuum was monitored by a heated thermocouple junction. The circuit shown in Figure 1 provided the necessary signals. The heater current of 20mA was generated by the voltage regulator configured into a current source. The thermocouple junction voltage was detected and amplified to a level acceptable to the telemetry. The H.V. monitors are also shown in the same figure.

4. Pulse Detector

The output pulses of the electron multiplier were conditioned by the circuit shown in Figure 1. The charge sensitive preamplifier-discriminator (AMPTEK A-101 PAD) was set to detect charges of 10^{-12} Coulomb. The preamplifier output pulses of 220ns were conditioned by the flip-flop for transmission to the pulse counter in the flight control unit.

5. Sweep Circuits

The digital circuits to control the quadrupole excitation signals are shown in Figure 3. The code designating the start of an amu sweep was latched into the presettable counter (Z_{25} - Z_{27}). The counter was advanced at regular intervals until the last amu code in a particular sweep was

reached. That code was stored in latches Z_{28}, Z_{29}) for comparison with the output of the counter. Upon a match the comparator (Z_{31} - Z_{33}) produced a pulse used to initial the next control process. One of the possible processes adjusted the ion count for system noise. During this time the ratio between the ac and the dc qudrupole excitation signal components was raised to inhibit the ions from reaching the electron multiplier. This was accomplished by putting the latches Z_{29} and Z_{30} containing the ratio code into the high impedance state. The amu and the ratio control codes were periodically transferred into the shift register (Z_{34} - Z_{36}) for transmission to the PCM encoder.

The digital amu and ratio control codes were converted into appropriate analog signals by the DAC's and the amplifiers shown in Figure 4. Thus, the analog signals for the DC amplifiers and the control of the RF oscillator were generated. The additional gain necessary for operating the quadrupole mass filter in the high pass mode was provided through the FET gate.

To obtain the required DC voltage levels and to inject the common 0 bias pedestal into the quadrupole structure, circuits shown in Figure 5 were used. High voltage operational amplifiers augmented by power transistors (Q_3, Q_4) were employed to handle the large voltage range. Current sources (Q_2. Q_6) were used in the collector circuits to minimize power dissipation during quiescent periods. To decrease the rise time during large signal increments the current source capability was increased from 3mA to 25mA by forcing Q_1 and Q_5 into saturation for a short duration.

6. The AC Exciter

The circuit diagram of the oscillator which provides the ac excitation signal to the quadrupole filter is shown in Figure 6. The transformer

driving the 1.9cm quadrupole rods was wound on an acrylic toroid 11cm in diameter, 5cm high and a wall thickness of 1cm. Chokes were used to prevent the ac from entering the dc excitation signal circuits.

The oscillator operated at 600kHz with the peak voltage spanning a 10 to 1600 volt range. The control over the amplitude was exercised through an operational amplifier (3581) where the control and the feedback signals were summed. The feedback signal was derived from the transformer output winding through a capacitive divider and a dc restorer circuit.

To provide sufficient ac feedback signal for the driver transistors (2N5008) of the oscillator, additional capacitors (3.2kpF) were switched through VN10KM into the base drive circuits during the operation in the lower output range. The switchover, in the excitation signal controlled through the bias control circuit, produced a transient of short duration. Therefore, the point where the feedback circuit switched, was selected to fall within a range of the mass spectrum of little or no interest to the experimenter.

To protect the oscillator from accidental overdrive or overheating due to a prolonged operation in the upper mass range, current sensing and thermal shutdown circuits were used. Both circuits provide abrupt recovery to insure resumption of oscillation.

C. THE FLIGHT CONTROL UNIT

The Flight Control Unit (FCU) provided digital signals to control the operation of the ion mass spectrometer. Commands and data also were processed by the unit. The control unit consisted of two subsystems. The primary system was based on an 8085μP while the secondary system employed

CMOS combinational/sequential logic. The control of the instrument could be transferred between the two systems through the tone command link of the balloon. Both systems worked through a common interface to reach the digital to analog conversion circuits which controlled the generation of the excitation signals for the mass filter and the bias voltages for the ion optics.

1. CPU

The μP based control circuits are shown in Figure 7. Only two interrupts were used to divert the μP to priority tasks. Request for data from the PCM encoder utilized the interrupt 7.5. The RF reset command was utilized to activate the TRAP interrupt in order to return the system to the beginning of the data gathering program without destroying the elapsed flight time counter in U143 (CLK2).

Normalization of the spectrometer data to counts per second before the transmission from the RAM was performed by the arithmetic unit (U141). The unit was not capable of operating at the μP clock rate. Therefore, the clock frequency was halved and a WAIT state was generated by U163 and U170 respectively.

Data from the mass spectrometer was received by the counter in U143 (CLK1). The output of U158 was reset to ZERO at the beginning of every data collection period by a signal originating at PA3 of U142. The status was sensed at PCØ of the same unit. This process in effect reset the flip-flop in the data conditioning circuit at the electron multiplier. Upon command the data counter in U143 reset itself on the negative transition following the first positive transition of the input. Only then the counting began. Therefore, the registered count could differ by as

many as three counts from the actual number of ion impacts. To correct the result U166 was used. At the end of the data collection period the outputs of U158 and U166 were examined. A ONE at the output of U166 resulted in an addition of two to the count. One was added when the output of U158 was found to be high. Error of three counts was indicated when the outputs of both circuits were high.

The other two counters within the U143 were used for timing purposes. The elapsed time in seconds since the last reset of the whole system was kept by the counter 2. The length of the data collection period was determined by the counter Ø. The two counters were driven at 1Hz and 200Hz respectively. The signals were derived from 204.8kHz clock (BΔ) through U168, U169 and U172. The end of the data collection period was transmitted to PC3 of U142 and to the data conditioning flip-flop in the mass spectrometer. The data collection began when the inhibit signal at GATE 1 of U143 and MR of U169 was removed. The counter Ø of U165 determined the frequency of the AC exciter. This information was used to adjust the amplitude of the quadrupole excitation signals to compensate for frequency drift. The counter was activated periodically for one second by U174-U176 and PA2 of U142. The frequency correction subroutine could be bypassed through the switch at PB1. The counter 1 of U165 timed the length of communications through the serial command link. Any attempt to communicate beyond allotted time was interrupted through PC5. The UART (U159) was driven by a clock signal generated in U142. To insure that communications were attempted through a viable link, the AGC of the balloon borne receiver was connected to $\overline{DSR}$ of the UART.

Other control signals passing through the U142 I/O ports included the RAM DUMP request to and the acknowledgement from the PCM encoder (PA0 and PC1 respectively). The request to dump the RAM information was inhibited during combinational logic operation (U164). The synchronization of the data to PCM encoder was accomplished through PA1, PC2 and U163. Finally, the message from the tone command link not to adjust the mass spectrometer data for noise induced errors was received through PB0.

2. Buffer and Interface Circuits

The address bus to the RAM and the EPROMS' was buffered through U145, U146, U152, U153 shown in Figure 8. The data bus to the RAM/EPROM CIRCUITS was split into two unidirectional buses buffered by U151 for the outgoing and U167 for the incoming data. U144, 147, 148, 149, 162 and 177 buffered and created various chip select, enable and strobe signals to the memory circuits. The interface to the CMOS data circuits was created by pull-up resistors (U150) and CMOS buffers U154 and U155. Low power Schottky TTL (U161, 162) provided chip select signals. The CMOS latch U156 transmitted control data to the D/A interface circuits. A code consisting of the 5LSB's of U160 was decoded in the combinational logic circuits where strobe signals were generated to latch the control data into the appropriate circuits. The MSB was used to start the mass spectrometer data collection process. U157 served as a temporary storage for data to the PCM encoder. Finally, the U164 buffered the CPU/COMB LOGIC control selection signal from the tone command circuits.

3. Memory

The CPU programs and the mass spectrometer flight control library were stored in six of the eight 2716 EPROM's (U98-U103) shown in Figure 9. Two were used as spares. Units U198, 199 were assigned to the

CPU, the rest to the spectrometer.

Only 4k bytes (U90-U97) of the 16k byte RAM used as a temporary data storage are shown. Since the memory was located on a separate board from the MPU, the two data buses (DBIN, DBOUT) and the address bus were buffered by U206-U209. Three to 8 decoders (U204, U205) provided the chip select signals.

4. Combinational/Sequential Logic Circuits

Figure 10 shows the timing and the data circuits of the combinational/sequential portion of the MS flight control unit. The length of a data collection period (dwell time) at a given mass domain, as well as, the duration of the data adjustment process for noise induced errors was controlled by these circuits. The data collection circuits and the shift registers necessary to present the data in a proper sequence to the PCM encoder also were included in this portion of the system.

The clock signals originated at the oscillator formed by U86 and associated components. The 3.2768MHz output was converted into two signals of 0.2048MHz and 0.8192MHz by the frequency divider U85. The former signal was transmitted to the CPU section as the BΔ time. The latter signal clocked the presettable divide-by-N counter (U65, 66,88 and 89). This presettable frequency divider in conjunction with the counter U54 established the dwell time in multiples of 5 milliseconds. The divider was preset to a count originating in a set of instructions controlling the ion mass spectrometer during a given segment of a program. Latches U56 and U91 served as a temporary storage for that 16 bit dwell time determining instruction.

A 16 bit data counter was formed by U61-U64. The data (BQ) entered the counter through the circuit of U78. To correct the count for the

division by two, performed in the signal conditioning circuit at the electron multiplier, the contents of the counter were shifted one position towards the MSB when transferred into the serial-to-parallel converter (U50, 51). The least significant bit of the incoming data stream was introduced directly into the S-to-P converter. This accounted for odd numbers of ion impacts.

To adjust the collected data for noise induced errors the counter was set into the countdown mode. The data signal triggering the counter was inverted. During the correction process the ions were prevented from reaching the electron multiplier. Thus the data count was reduced by the number of noise induced multiplier pulses. Obviously the same dwell time as for data collection was used for the adjustment.

During the switchover into the countdown mode, the contents of the data counter were preserved by a preset enable signal from U72. This "store the count" command originated in the control section of the CL system.

The counting process started with a reset of the dwell time counter which also enabled the data counter. A GO command at U77 enabled U83. The dwell time counter started receiving clock pulses. At the same time the flip-flop in the electron multiplier section was enabled. An overflow in the U54 marked the end of the data collection interval by inhibiting the flow of data. A DATA READY pulse was generated by U79.

The circuits U50-U53, 55, 71 and 74 formed a portion of the parallel-to-serial data converter. Other circuits of the chain were located in different parts of the mass spectrometer electronics package. The interconnections with the other members of the chain are indicated by letters B, L and M. The parallel-to-serial converter was configured to present,

upon request, the various digital data and monitor signals to the PCM encoder in a prearranged sequence.

The data and the dwell time registers (U50-U53) were loaded by the P/S control pulse. That pulse was generated only when new data was ready for transmission. The registers U55, 71 and 74 used to indicate system status were loaded by 1st. FRAME signal for each minor PCM frame.

The shift registers were clocked by a signal from U67. Upon request from PCM encoder (MS of U79) the gate U73 was enabled allowing pulses from the oscillator U76 to clock the counter U67. The flip-flop U77 was SET after eight clock pulses were passed to the shift registers. Thus a new byte of data was shifted into U68. Buffer U70 presented that data in parallel form to the digital multiplexer of the PCM encoder.

The control signals for the mass spectrometer originated in the circuits shown in Figure 11. Each segment of the flight program was defined by a 32 byte instruction set. These instructions were stored in the EPROM's U3 and U4. Sequential selection of each instruction within a set was controlled by the counter U7 and U8. The same 5 bits that addressed the EPROMS also controlled the 1 of 32 decoder (U12 and U13) used to generate strobe signals to latch each instruction into an appropriate D to A interface circuit or to control events within the combinational/sequential logic circuits. The width of the strobe pulse was determined by U15 which enabled the selection circuit for a fraction of the system clock period. Units 44, 45 and 36 buffered the strobe signals, while U11 was used as a tri-state interface between the CL circuits and the MPU signals (SELX).

The starting addresses of the instruction sets were stored in U2. A counter (U1) provided the 7LSB's of the address. The 4 MSB's were provided by the presettable counter U41. The counter could be preset through the tone link. Thus up to 16 different programs were available in the

flight repertoire. The selected instruction sets were presented in parallel form to the mass spectrometer circuits by the buffer U5.

Control over the sequence of events within the CL control unit was exercised through a sequencer consisting of a presettable counter U29 and 4 to 16 decoder U28. Gates and flip-flops routed and stored the control signals to and from other parts of the unit.

When a new instruction set was to be introduced into the mass spectrometer circuits, a ONE at Qo of U28 inhibited the counter U29 through U18, 21, 27 and 37. The circuit consisting of U16, 18, 19, 42, and 33 was enabled. Thus the clock pulses from the oscillator U17 were allowed to reach counters U7 and U8. Once the set of 32 instructions had been loaded, a strobe signal from the 1 of 32 selector (15) again enabled the sequencer counter through U21, 22, 24, 27 and 37. A ONE at Q_1 of U28 generated BINFO RST to clear the data counters. The GO command to start the data collection was given when the Q_2 output was selected. The sequence counter (U29) was inhibited until the DATA READY signal indicated the end of the data collection interval. Next, the command to store the dwell time was generated at Q_3. A conditional jump was executed when the sequencer reached Q_4. When the data correction for noise induced errors was not required, the counter U29 was preset through U25 to select Q_9. This occurred when a NO DOWN command was received and/or the data count exceeded 256 (Q_7 of the data counter was set). When the sequencer reached Q_{10}, DATA VALID signal was generated and the sequencing stopped until the FS signal from the PCM encoder was received. Upon arrival of the FS signal the 1st. FRAME and the P/S pulses loaded the parallel-to-serial data conversion registers. When the sequencer advanced to Q_{11} the INC DOM

pulse incremented the mass filter by one quarter of an amu domain, provided the MSB of U2 was set. At the same time FF U26 was reset. The FF served as a flag to indicate that data were to be collected at the same mass filter setting, but with a different set of bias voltages. That FF was set every time the loading of the second set of bias parameters has been completed, indicating that for the next data collection period the amu domain must be incremented. The FF was reset every time the mass filter was incremented. Thus the system was automatically prepared to collect data at the same filter setting, but with another set of bias voltages. The MSB of U2 overrode that command and reset the sequence counter. Otherwise a pulse at Q_{12} initiated the loading of the second set of the bias parameters, while Q_{13} stopped the sequencer until the loading was completed. Since the amu information had to be preserved the counter U7 and U8 was preset to 1001. Thus the locations containing the amu settings in the EPROMS were bypassed. When the loading was completed a pulse at Q_{14} reset the sequencer to the Q_0 state for a new cycle. If the loop counter U9 and U10 had not reached ZERO during the last control cycle, the same program was repeated. Otherwise a new program was run.

When data correction for noise induced errors was required the sequencer proceeded from Q_4 to Q_5. The command to store the count was sent to the data counter. Q_6 reset the UP/DOWN control, Q_7 enabled the data counter and Q_8 stopped the sequencer until data ready signal was received. From there the sequencer proceeded through the steps described in the branching sequence.

5. Tone Command Conditioning

Eleven tone command link channels were assigned to control the instrumentation associated with the mass spectrometer. Circuits shown in Figure 12 were used to condition the commands into appropriate control signals.

Relay ground closures were the outputs of the tone command link decoder. Debounce circuits (U1, U17) conditioned these signals for further processing. To control latching relays in the mass spectrometer circuits, the command signals were split into two outputs providing ON/OFF pulses on each alternate command received through the same channel. Steady state level commands were also available. This signal conditioning was accomplished in the circuits of U3 through U12.

Two of the tone command channels were assigned to select one of the 16 programs available to the mass spectrometer while under the combinational logic control. One channel transmitted data. The other was used to strobe each data bit into a shift register (U13, U14). Only ZEROS had to be transmitted preceding the strobe command. Otherwise a ZERO was shifted into the register. Each four bit selection code was preceded by an 8 bit identification word. Only when the comparator (U15, U16) detected the identification word, the four program selection bits were accepted. Then the CD VALID pulse together with the four bit code were transferred to the combinational/sequential control circuits.

6. PCM Encoder

The PCM encoder was designed to accommodate 48 analog and 10 eight bit digital signals. One of the digital channels was dedicated to the mass spectrometer data. The format of the PCM signal and the selection

of the various inputs for precessing was controlled by a program residing in an EPROM. A complete description of the operation and the capabilities of the programmable encoder may be found in Reference 3.

The circuits of the PCM encoder may be separated into two functional parts for convenience: the control signal generator and the data processing. The circuits shown in Figure 13 provided the timing and the control signals to the data acquisition and processing components. The clock for the parallel to serial data converter (U6, U7) was generated by units U1-U3 and U9. A bit rate of 12kb/s or 48kb/s could be selected through the balloon tone command link (fc). Another clock signal at twice the selected frequency was used to generate interval timing signals. Two decoded counters (U4, U5) driven by that clock synchronized the control sequence. During the second and the third bits of each word within the PCM data stream a program counter (U14 or U19) was advanced twice. The first control byte stored in an even numbered address location of the EPROM (U12) was latched for temporary storage into U11. The second byte remained available on the output lines of the EPROM. The control pulses for this sequence originated at Q_2-Q_4 of U4. After this sequence was completed, the control circuits were deactivated until bit 8 of the data word. During the last bit the data available on the data bus (DBX) was transferred into the P/S converter and the two control bytes were latched into U16 and U17. The control byte in U16 was used to control the analog data multiplexer signals on lines (CS1-CS3, AS1-AS4) or to signal to the user of the digital channels that the encoder was ready to accept data (SR1-SR6). The 4 LSB's of the control byte in U17 selected the data words to be inserted into the PCM train. The 3 MSB's were used to format the output of the encoder. A ONE in

the MSB position signified an end of a minor frame. In conjunction with a pulse at 3 of U4 (during the first data bit of a PCM word) it reset the minor frame counter (U14), advanced subframe identification counter (U29), acknowledged "RAM DUMP" request from MPU (13U30) and reset FF 1U30 thus insuring that an even numbered control byte was available at the output of the EPROM. ONE in the NMSB position accessed the subframe program through the proper selection of the tri-state buffer circuits (U13, U15, U18). It disabled the minor from program counter (U14) and enabled the subframe program counter (U19). The end of a major frame was controlled through a ONE in the 3rd. MSB position during the last word of a subframe. A pulse generated at 10U22 during the last word of the minor frame reset the subframe program and the identification counters (U29).

Other units (U10, U24, U26) were used to provide sync signals to an analog to digital converter and for diagnostics. U27 could be used to extend the minor frame count to 12 bits during the "RAM DUMP" mode. The count could be used for identification of data blocks transmitted from the RAM.

The data circuits are shown in Figure 14. Two of the 16 channels of the digital multiplexer (U1-U8) were wired to produce the 16 bit frame synchronization pattern. Mass spectrometer data (MS), subframe identification code (S), ADC data (AN and AN9-AN12) and the ONE's COUNTER (BC) data occupied additional channels. The counter was used to determine the number of ONE's within the minor frame. The frame synchronization words and the count it self were excluded. That word could be utilized as an indicator of transmission errors within the frame. The other 9 channels were assigned to the digital data buffered by U15-U26.

The digital data from the 12 bit ADC could be transmitted as the 8 MSB's only or as the full 12 bit word utilizing the 4 MSB of the adjacent word in the PCM pulse train. The same process was used to extend the range of the minor frame counter during the "RAM DUMP" mode.

The ADC converted data selected by the multiplexer (U9-U11). Only few of the analog channels were used to monitor mass spectrometer functions. The rest were assigned to other instruments of the scientific package. A 4 pole active premodulation filter and two temperature sensor amplifiers completed the data conversion package.

D. THE GROUND CONTROL UNIT

The Ground Control Unit (GCU) was developed as a specialized stand alone command, control, communications and monitor interface between the operator and the ion mass spectrometer during development and laboratory testing. During the airborne operations radio links to and from the balloon instrumentation had to be provided. Interface to TTY or CRT terminals also were included.

Single stroke commands entered on a key pad were presented, upon request from the airborne unit, to the serial command transmitter through an RS232 interface. Responses from the flight unit were received either through the serial down link or through the PCM data stream. The viability of the communication link was checked through the AGC signal from the receiver.

The PCM data was accepted from the telemetry receiver and demultiplexed, provided the clock from a bit synchroniser was also available. Any received data or monitor word could be assigned to one of ten DAC's or to any one of four digital displays. Outputs were provided for the

analog signals to be used with recorders and/or oscilloscopes. Eight microammeters were also provided to monitor the performance of the mass spectrometer. The demultiplexed data was available, one word at a time, for other equipment. Programming of the demultiplexer for a given PCM format and output channel assignment was aided by prompting words or phrases appearing on an alphanumeric 16 segment display.

The unit was contained within a small suitcase type instrument box and required only a 28 volt, 800mA external supply.

The circuits of the GCU may conveniently be subdivided into three sections. The section composed of the 8085 microprocessor and supporting circuits controlled the unit; USART's, I/O ports and DAC's handled the communications and data, while the keypad and the LED displays interfaced with the operator.

The control section is shown in Figure 15. The EPROM's Z_{11} through Z_{13} stored the programs to control the operation of the GCU. The lower byte of the address to the EPROM's appearing on the multiplexed bus of Z_1 was stored in Z_{15}. A 4k byte RAM was formed by the 1k x 8 bit static memory chips Z_7 - Z_{10}. The bus line A10 in conjunction with Z_{21} selected the appropriate memory circuits. The data and the address bus lines to a temporary memory in the combinational/sequential flight control circuits were buffered by Z_{17} and Z_{18}. Strobes were provided by Z_{29}. The temporary memory plug-in unit was used in place of EPROMS during the development and laboratory tests.

The communications with CRT or TTY terminals and with the mass spectrometer were carried through USART's Z_5 and Z_6 respectively shown in Figure 16. RS232C interface units Z_{24} and Z_{25} provided the necessary

signal level translation. A triple 16 bit counter chip Z_4 provided the clock and timing signals for the two communications links. Unit Z_{23} was used to divide the 8085 clock frequency by two to accommodate the slower counter chip.

The incoming PCM clock and data were processed through the Schmitt trigger buffers Z_{85} and the serial to parallel converter Z_{30}. Data collected in that register was transferred onto the bus through the port PA of Z_3. The interrupt for the data transfer was generated in Z_{28} after each group of eight clock pulses. It signified a reception of an eight bit data word. Frame detection and word synchronization was done by software. A word synchronization pulse reset Z_{28} through port PC of Z_3. Port PB was used to present the received PCM data word to any external users. A strobe signifying the availability of the PCM word was generated through the port PC.

The data was placed into two temporary storage sections of the RAM. From there the selected data words were transferred into the DAC's Z_{32} through Z_{41} or into the digital displays. Selection of the appropriate DAC was accomplished through Z_{22} and the Z_{20}. The latter also served as the selection of the I/O ports.

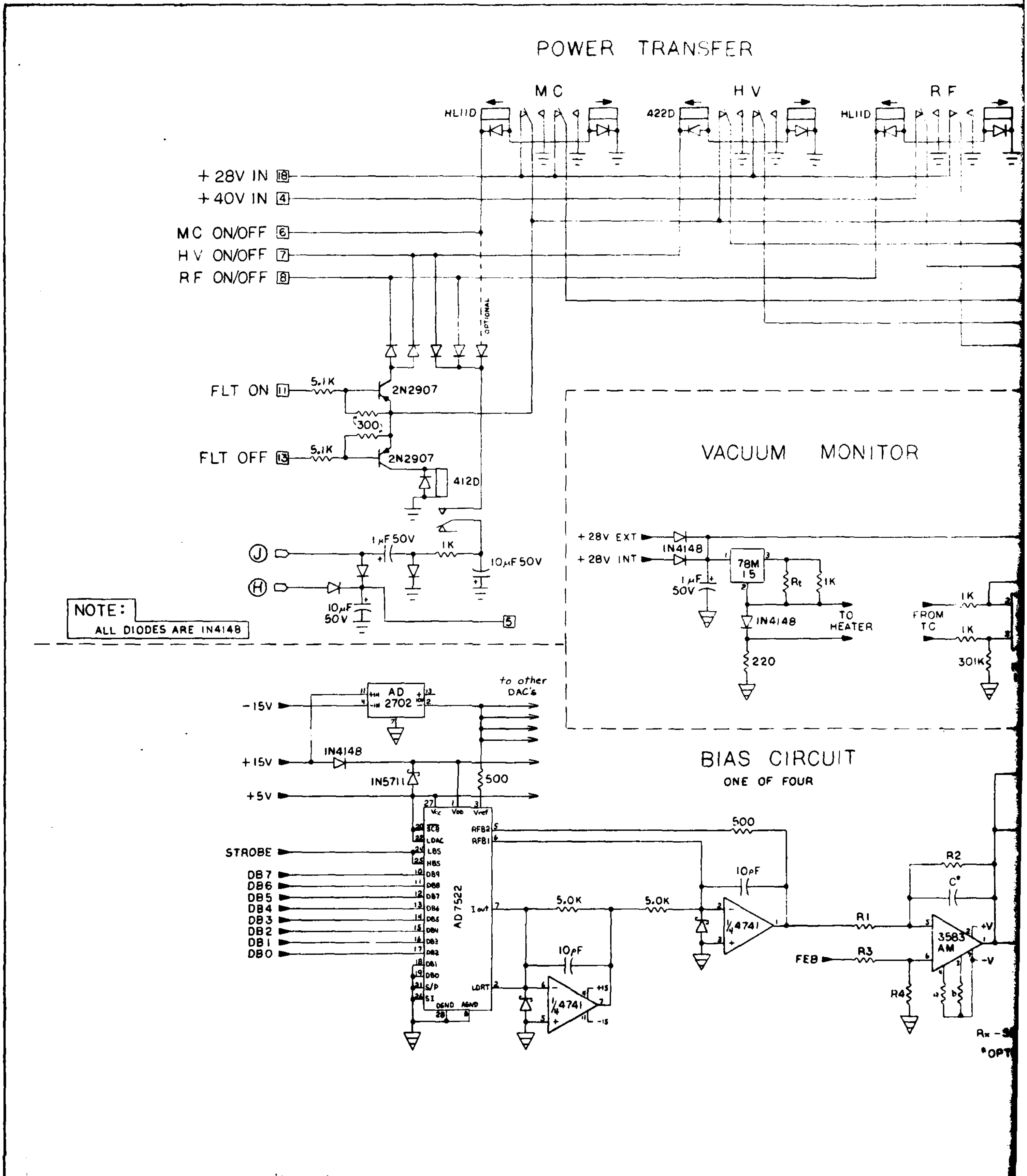
The programmable keyboard/display interface Z_{14} (Figure 17) in conjunction with the 4 to 16 line decoder Z_{43} and the multiplexer/demultiplexer Z_{42} scanned the keyboard and controlled the seven segment LED displays Z_{53} through Z_{68}. The eight character 16 segment alphanumeric display was controlled through a latching (Z_{16}) and a non-latching (Z_{19}) buffers and a binary to octal decoder (Z_{81}). The 16 bit code necessary to display one character were stored in two consecutive locations of the

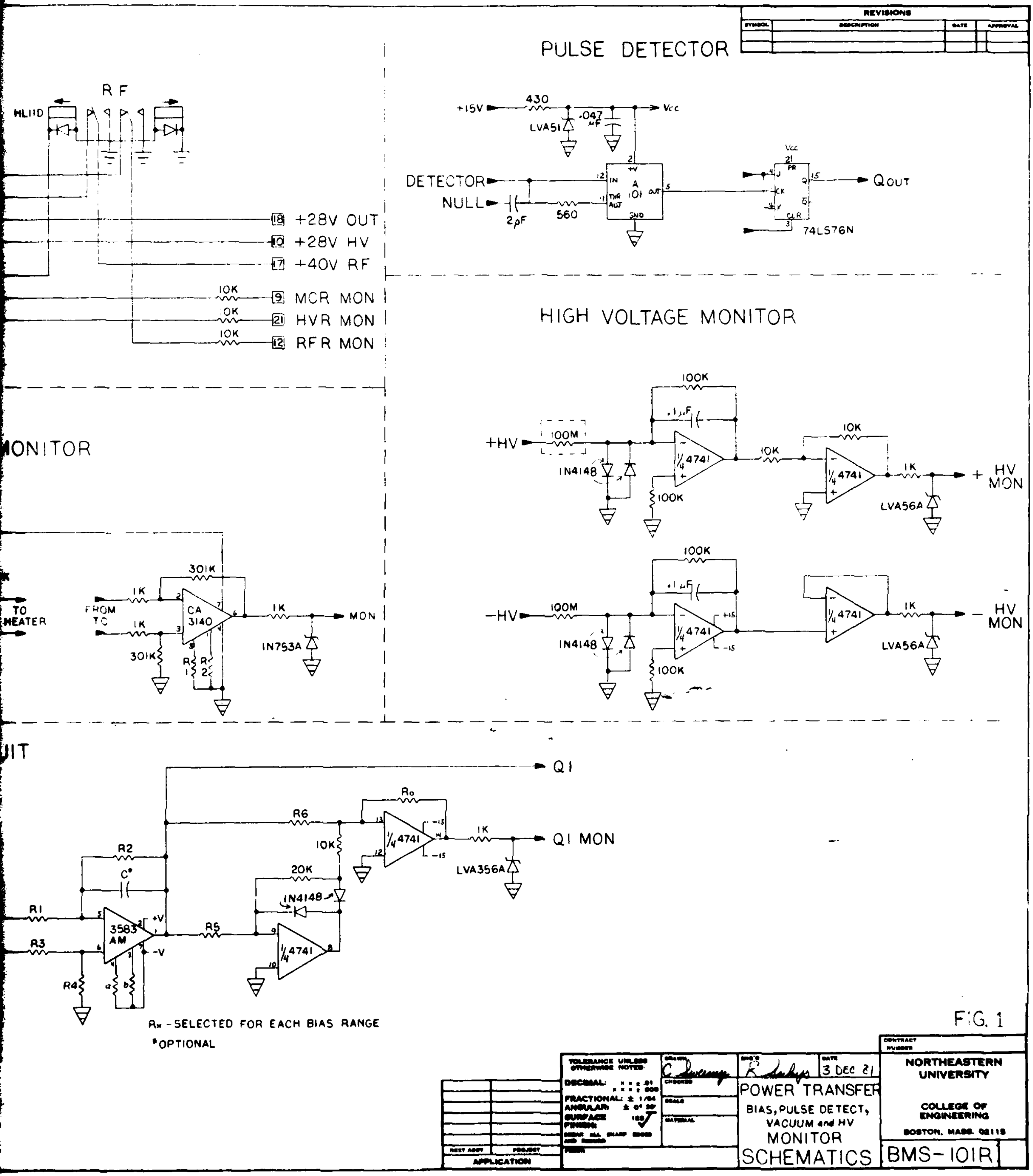
EPROM (Z_{72}). The addresses of the characters to be displayed were stored in the 256 x 8 bit RAM Z_{73} and Z_{74}. The addressing of the RAM for updating by the CPU was done through port A of the tri-state latch Z_{75}. Port B of that unit addressed the RAM when update of the alphanumeric display was required. Counter Z_{82} controlled the scanning of the display. Octal counter Z_{80} provided the strobe signal to latch the data into Z_{16} and advanced the RAM address counter (Z_{82}). The fast strobe pulses (2us) and the relatively slow (3ms) display period for each character were derived through the circuit of Z_{77} through Z_{80}.

The status of the entire system was indicated through an array of eight LED's. The PC port of the I/O circuit Z_2 was used to drive the display. The two remaining ports were kept as spares for future expansion of the capability of the GCU.

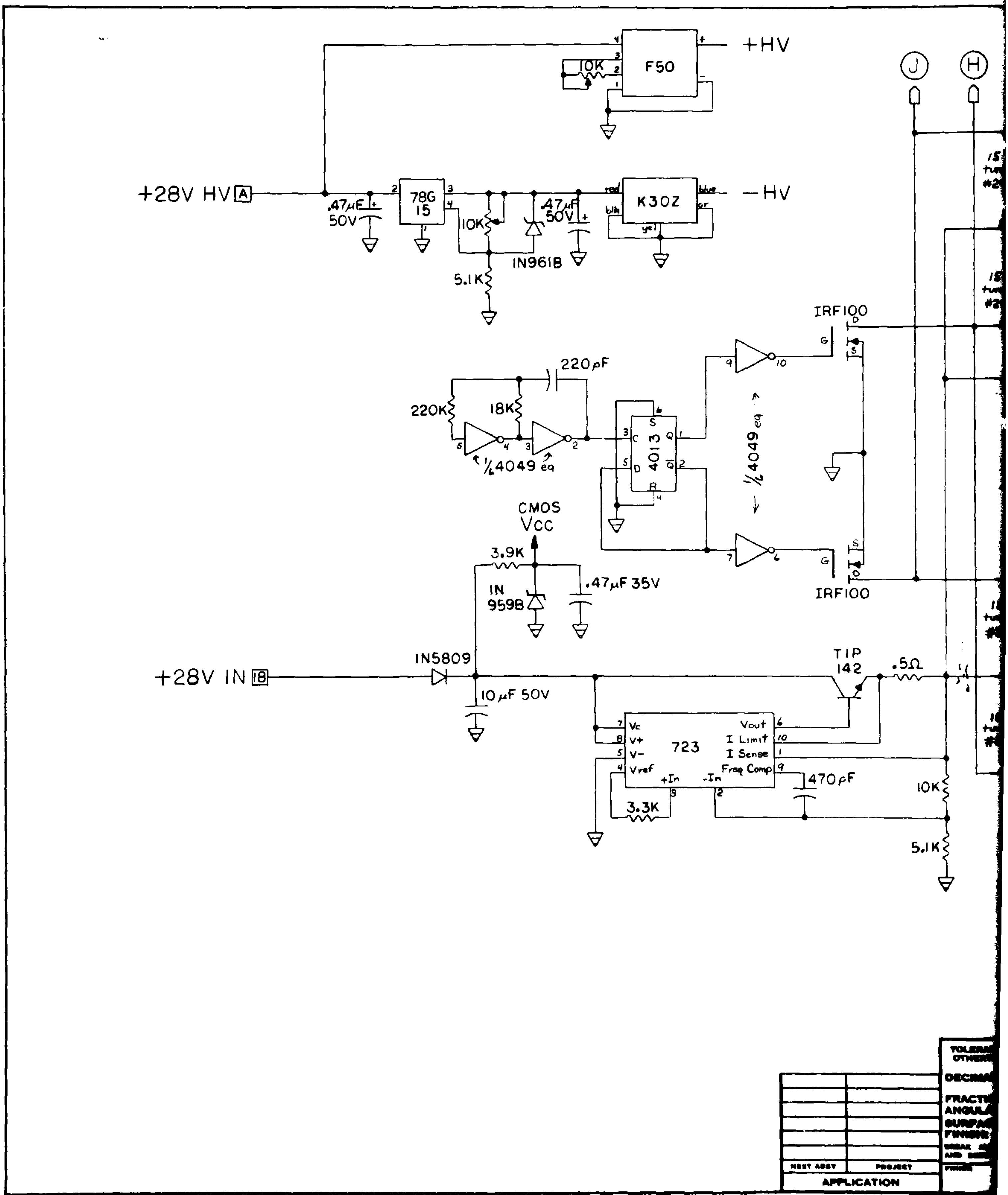
E. FLIGHT

The completed electronic package was delivered to AFGL in August, 1981. After testing and integration the field party left for the Balloon Launch Range at Holloman AFB, New Mexico on 1 September 1981. Contract personnel were not included in the field party. The flight took place on 29 September 1981 and was not considered successful from the standpoint of gathering scientific data. The gondola impacted the ground before becoming airborne, vacuum was lost during part of the ascent and the data stream terminated as a result of an apparent power failure. A complete analysis is not available at this time, but a modified unit will be designed and constructed under follow-on contract F19628-81-C-0162.


POWER TRANSFER
MC
HV
RF
HL11D
422D
HL11D
+28V IN 18
+40V IN 4
MC ON/OFF 6
HV ON/OFF 7
RF ON/OFF 8
OPTIONAL
FLT ON 11
5.1K
2N2907
300
FLT OFF 13
5.1K
2N2907
412D
J
H
1μF 50V
1K
10μF 50V
10μF 50V
5
NOTE:
ALL DIODES ARE IN4148
VACUUM MONITOR
+28V EXT
+28V INT
IN4148
78M 15
1μF 50V
Rt
1K
IN4148
220
TO HEATER
FROM TC
1K
1K
301K
BIAS CIRCUIT
ONE OF FOUR
to other DAC's
AD 2702
-15V
+15V
+5V
IN4148
IN5711
500
STROBE
DB7
DB6
DB5
DB4
DB3
DB2
DB1
DB0
AD 7522
500
10pF
5.0K
5.0K
10pF
1/4 4741
1/4 4741
R2
C*
R1
R3
FEB
R4
3583 AM
+V
-V


PULSE DETECTOR
RF
HL11D
+15V
430
LVA51
.047 μF
Vcc
DETECTOR
NULL
2pF
560
A 101
74LS76N
QOUT
+28V OUT
+28V HV
+40V RF
10K
MCR MON
HVR MON
RFR MON
HIGH VOLTAGE MONITOR
100K
.1μF
+HV
100M
IN4148
1/4 4741
10K
1K
LVA56A
+ HV MON
−HV
− HV MON
MONITOR
TO HEATER
FROM TC
301K
CA 3140
MON
IN753A
IIT
Q1
Ro
R6
R2
C*
R1
R3
R4
3583 AM
R5
20K
IN4148
Q1 MON
LVA356A
Rx - SELECTED FOR EACH BIAS RANGE
*OPTIONAL
FIG. 1
REVISIONS
3 DEC 81
POWER TRANSFER
BIAS, PULSE DETECT, VACUUM and HV
MONITOR
SCHEMATICS
NORTHEASTERN UNIVERSITY
COLLEGE OF ENGINEERING
BOSTON, MASS. 02115
BMS-101R
APPLICATION

2


+HV
F50
10K
J
H
+28V HV A
.47μF 50V
78G 15
10K
5.1K
1N961B
.47μF 50V
K30Z
red
blk
blue
or
yel
-HV
IRF100
220pF
220K
18K
1/6 4049 eq
4013
1/6 4049 eq
CMOS Vcc
3.9K
1N 959B
.47μF 35V
IRF100
1N5809
+28V IN 18
10μF 50V
TIP 142
.5Ω
Vc
V+
V-
Vref
+In
-In
723
Vout
I Limit
I Sense
Freq Comp
470pF
3.3K
10K
5.1K
NEXT ASSY
PROJECT
APPLICATION

REVISIONS			
SYMBOL	DESCRIPTION	DATE	APPROVAL

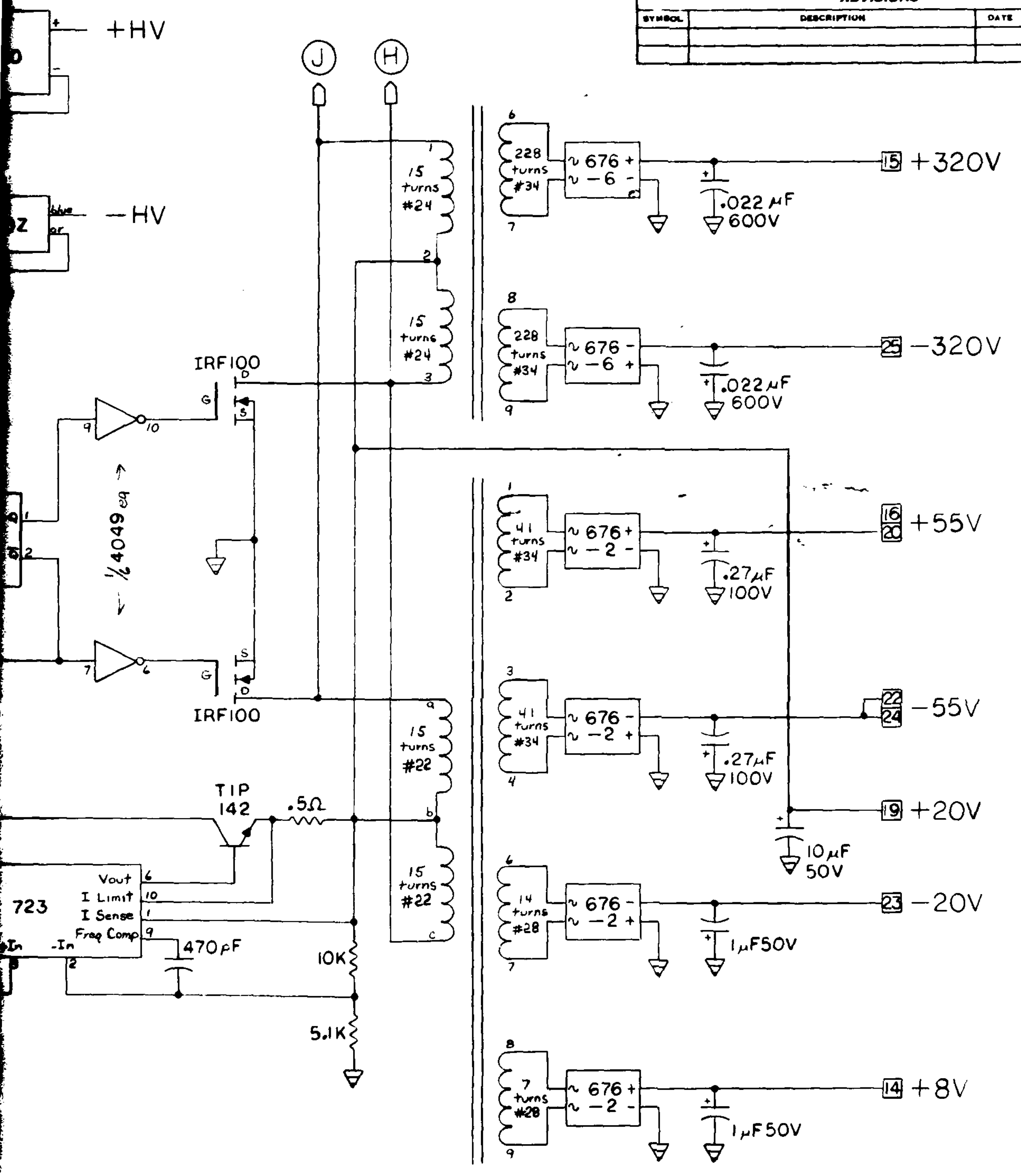


FIG. 2

NEXT ASSY	PROJECT
APPLICATION	

TOLERANCE UNLESS OTHERWISE NOTED:
DECIMAL: .XX ± .01, .XXX ± .005
FRACTIONAL: ± 1/64
ANGULAR: ± 0° 30'
SURFACE FINISH: 125
BREAK ALL SHARP EDGES AND BURRS
FINISH

DRAWN C Sweeney
ENG'R R Sakys
DATE 30 Nov 81
CHECKED
SCALE
MATERIAL

POWER SUPPLY SCHEMATIC

CONTRACT NUMBER

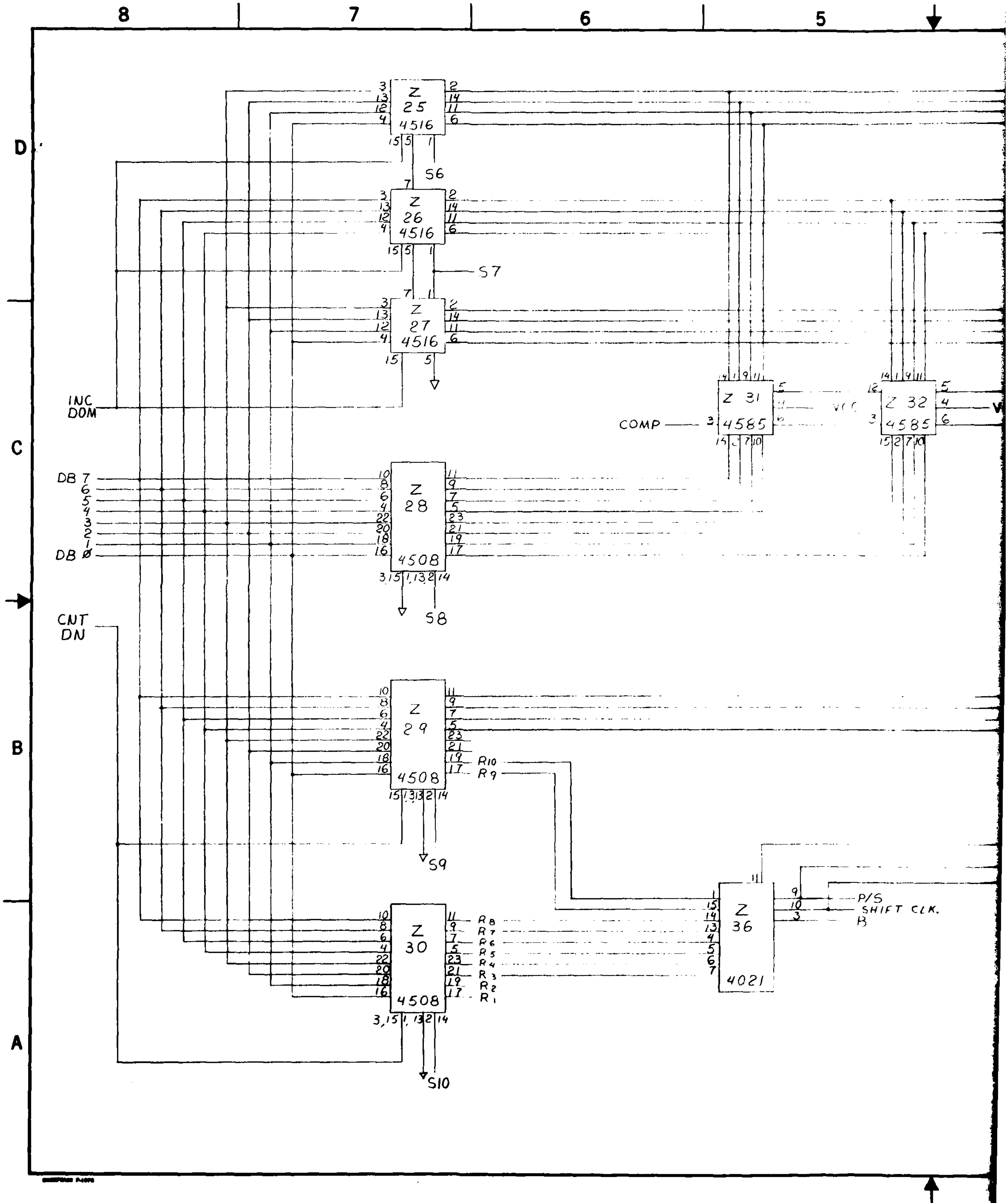
NORTHEASTERN UNIVERSITY

COLLEGE OF ENGINEERING

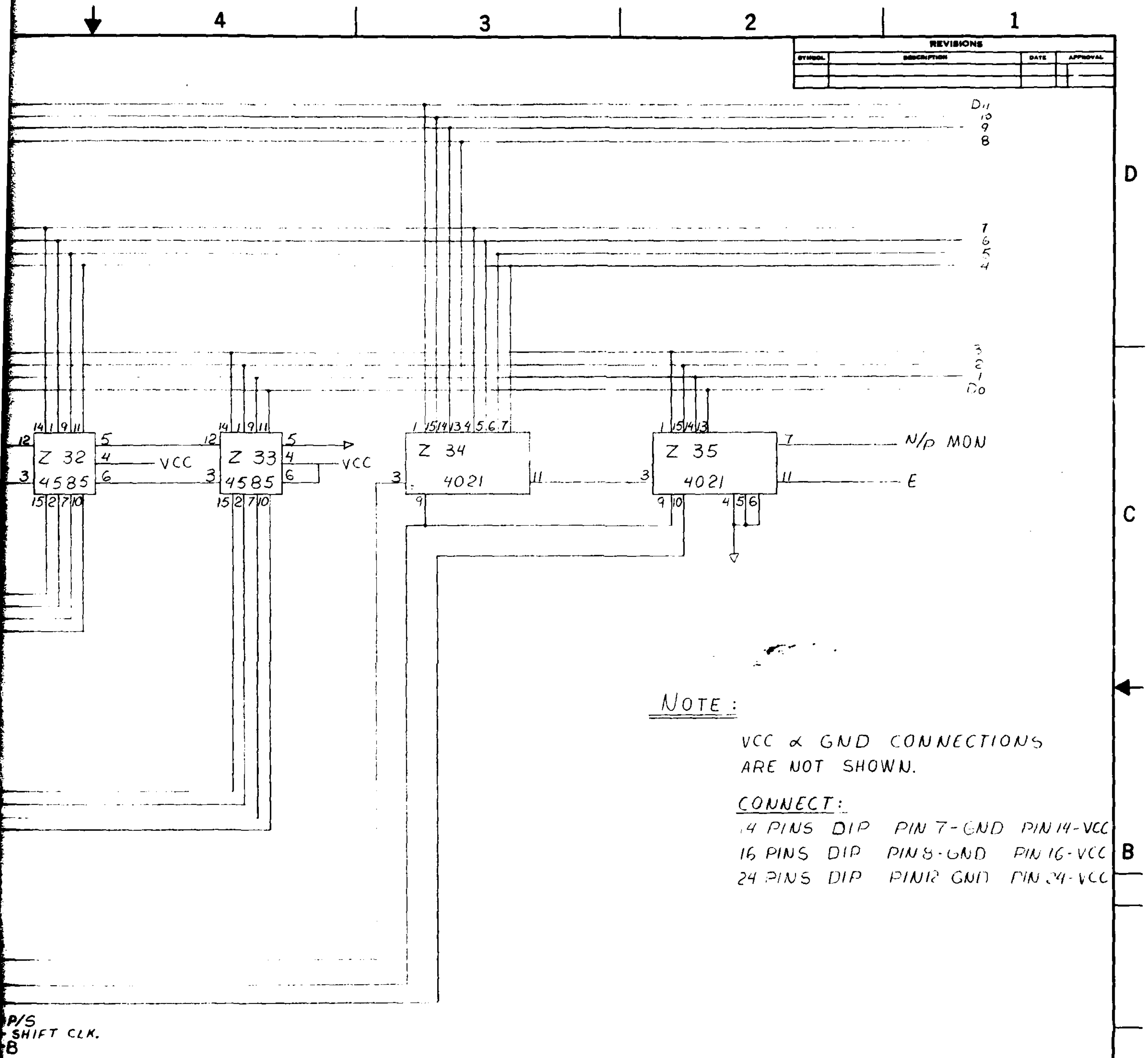
BOSTON, MASS. 02115

BMS-102R

2



8
7
6
5
D
C
B
A
Z 25 4516
S6
Z 26 4516
S7
Z 27 4516
INC DOM
Z 31 4585
COMP
VCC
Z 32 4585
DB 7
6
5
4
3
2
1
DB Ø
Z 28 4508
S8
CNT DN
Z 29 4508
R10
R9
S9
Z 36 4021
P/S
SHIFT CLK.
B
Z 30 4508
R8
R7
R6
R5
R4
R3
R2
R1
S10



NOTE:

VCC & GND CONNECTIONS ARE NOT SHOWN.

CONNECT:

14 PINS DIP PIN 7 - GND PIN 14 - VCC
16 PINS DIP PIN 8 - GND PIN 16 - VCC
24 PINS DIP PIN 12 GND PIN 24 - VCC

FIG. 3

TOLERANCE UNLESS OTHERWISE NOTED:
DECIMAL: .XX ± .01, .XXX ± .005
FRACTIONAL: ± 1/64
ANGULAR: ± 0° 30'
SURFACE FINISH: 125
BREAK ALL SHARP EDGES AND BURRS
FINISH:

NEXT ASSY | PROJECT
APPLICATION

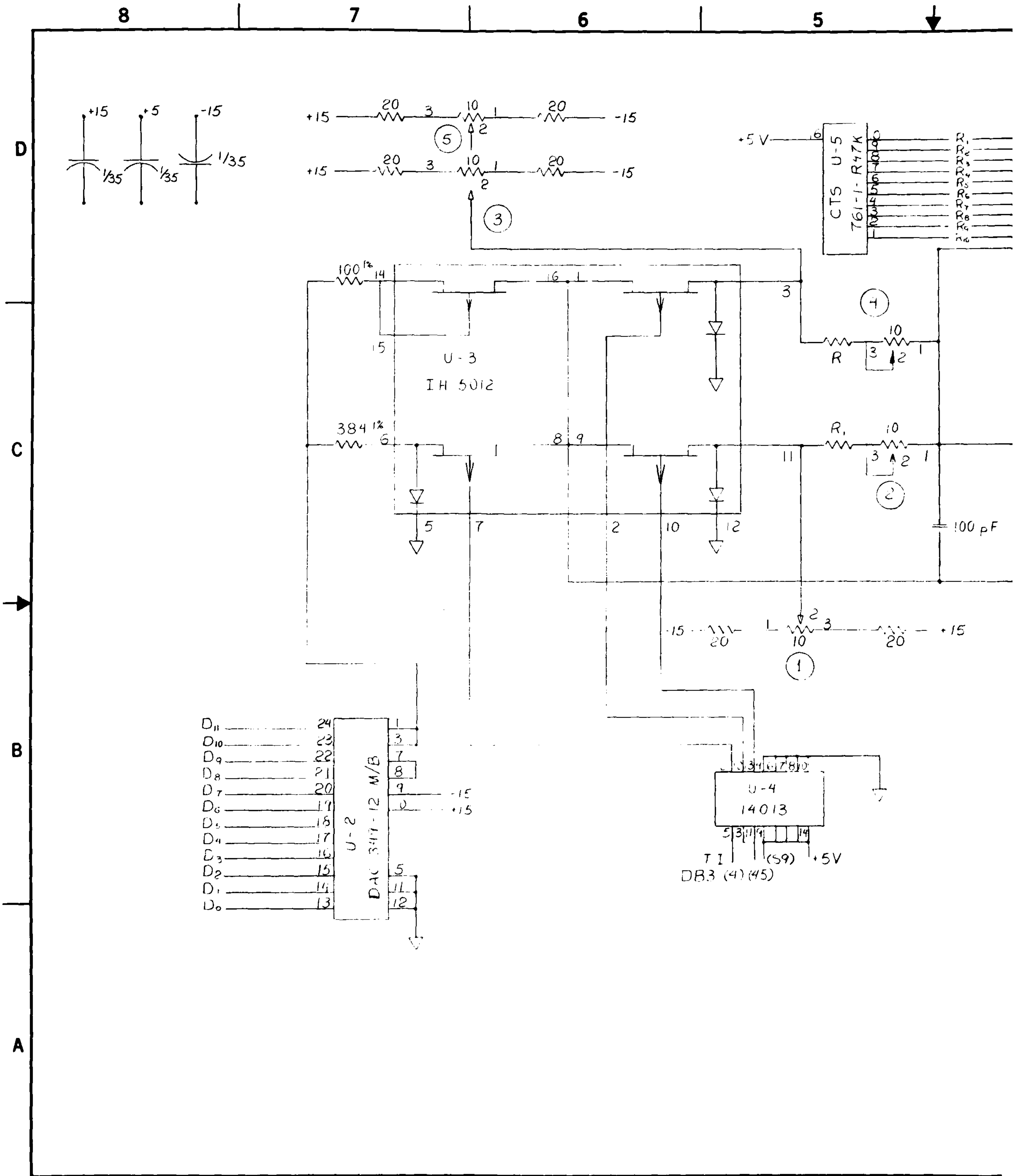
DRAWN: S. FARBER
DATE: 10-8-81

SWEEP CONTROL SCHEMATIC

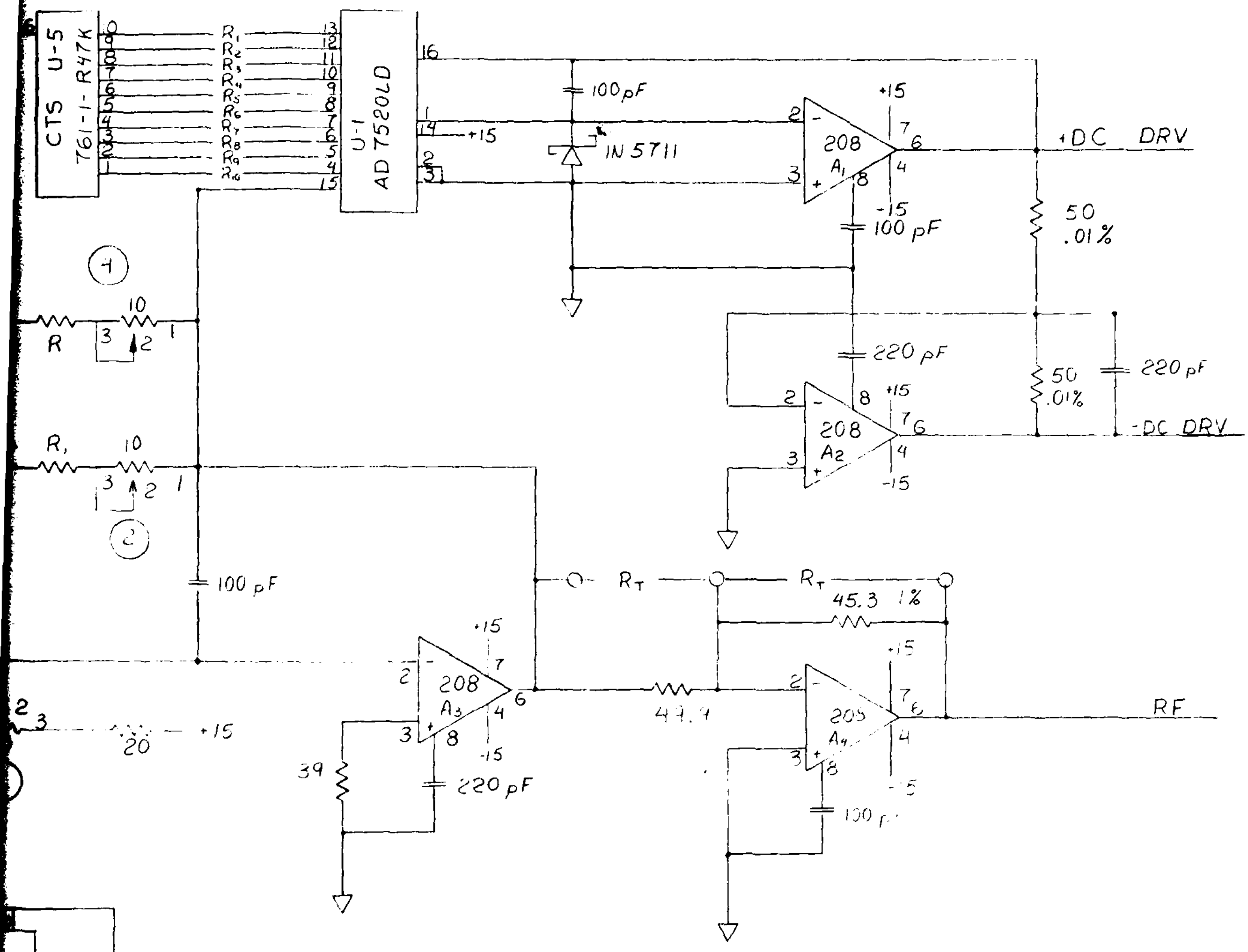
NORTHEASTERN UNIVERSITY
ELECTRONIC RESEARCH LAB
COLLEGE OF ENGINEERING
BOSTON, MASS. 02115

BMS-103R

2



8
7
6
5
D
C
B
A
+15
+5
-15
1/35
1/35
1/35
+15
20
3
10
1
20
-15
5
2
+15
20
3
10
1
20
-15
2
3
+5 V
6
CTS U-5
761-1-R47K
R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
100 1%
14
16
1
3
4
15
U-3
IH 5012
R
3
10
2
1
384 1%
6
8
9
11
R1
10
3
2
1
2
5
7
2
10
12
100 pF
-15
20
1
2
3
10
20
+15
1
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
24
23
22
21
20
19
18
17
16
15
14
13
U-2
DAC 349-12 M/B
1
3
7
8
9
0
5
11
12
-15
+15
U-4
14013
TI
(S9)
+5V
DB3 (4) (45)



NOTES:

RESISTORS IN KOHMS

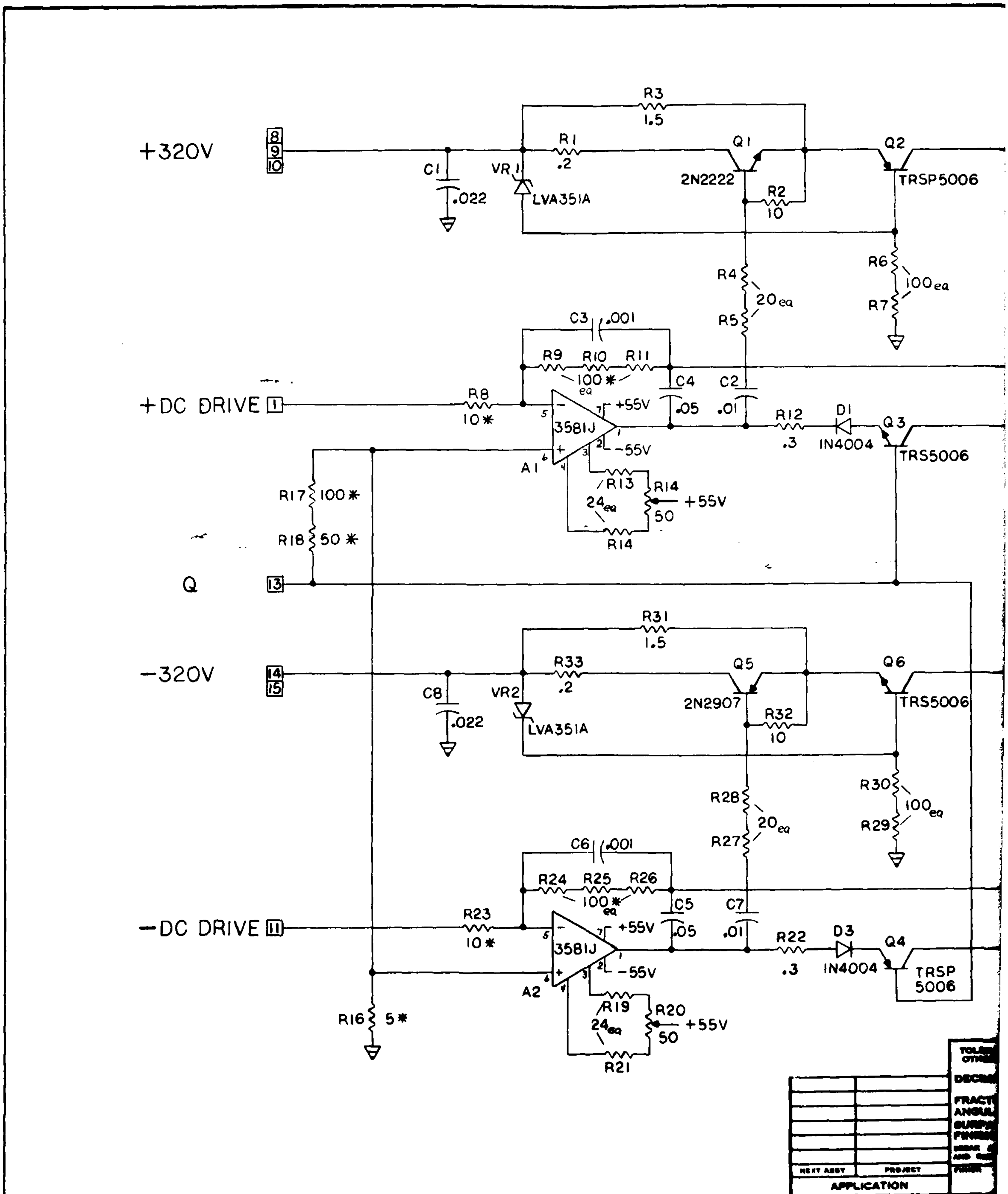
POTS - 25 TURNS

R - 82 K

FIG. 4

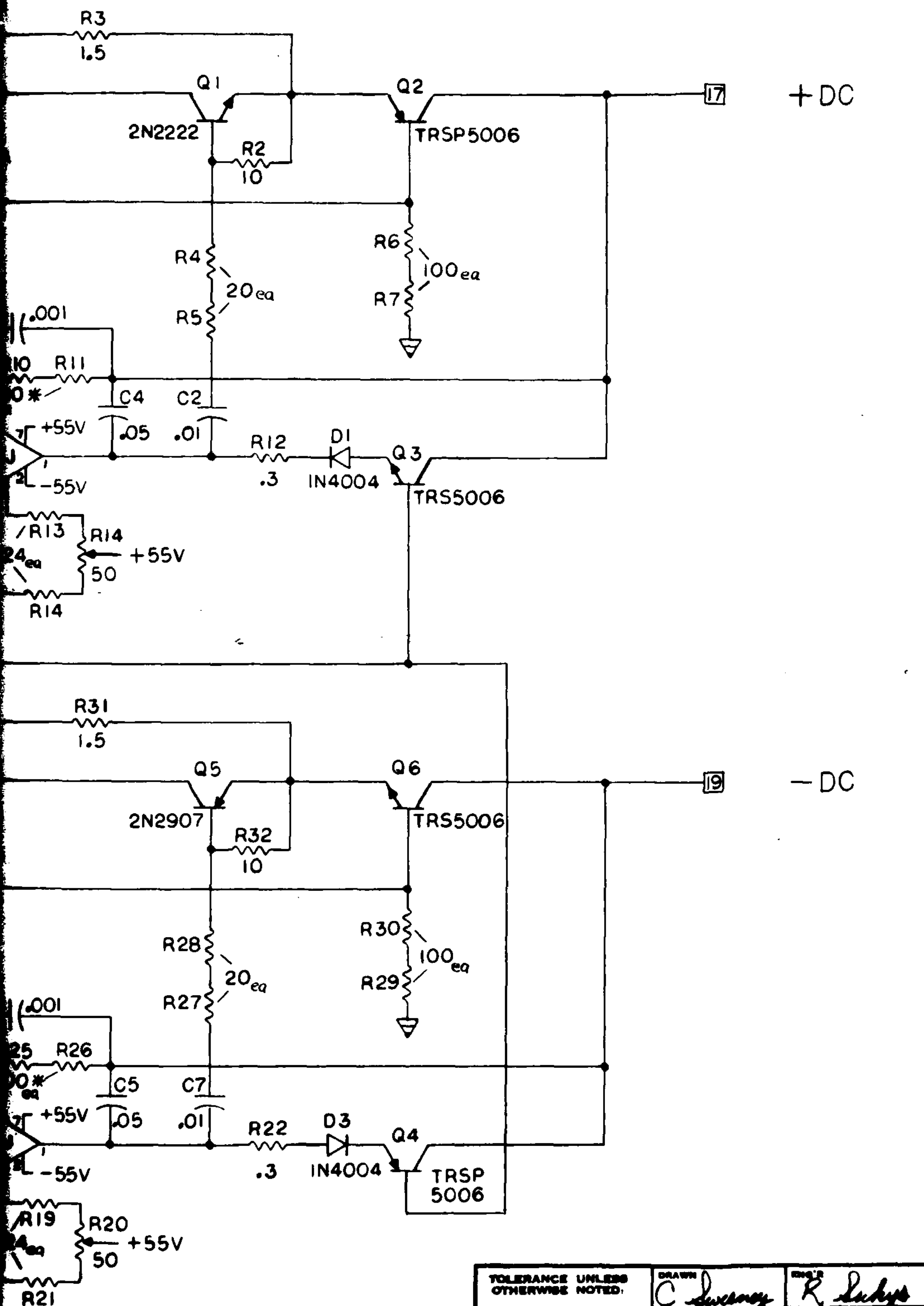
DRAWN	S. FARBER
DATE	10-20-81
Title	SWEEP GENERATOR SCHEMATIC
	NORTHEASTERN UNIVERSITY
	ELECTRONIC RESEARCH LAB.
	COLLEGE OF ENGINEERING
	BOSTON, MASS. 02115
Number	BMS-104R

2



+320V
8
9
10
C1
.022
VR1
LVA351A
R1
.2
R3
1.5
Q1
2N2222
R2
10
Q2
TRSP5006
R4
R5
20eq
R6
R7
100eq
C3
.001
R9
R10
R11
100 * eq
C4
.05
C2
.01
+DC DRIVE 1
R8
10 *
+55V
3581J
-55V
A1
R12
.3
D1
1N4004
Q3
TRS5006
R13
24eq
R14
50
+55V
R14
R17 100 *
R18 50 *
Q
13
R31
1.5
-320V
14
15
R33
.2
C8
.022
VR2
LVA351A
Q5
2N2907
R32
10
Q6
TRS5006
R28
R27
20eq
R30
R29
100eq
C6
.001
R24
R25
R26
100 * eq
C5
.05
C7
.01
-DC DRIVE 11
R23
10 *
+55V
3581J
-55V
A2
R22
.3
D3
1N4004
Q4
TRSP 5006
R19
24eq
R20
50
+55V
R21
R16 5 *
NEXT ASSY
PROJECT
APPLICATION

REVISIONS			
SYMBOL	DESCRIPTION	DATE	APPROVAL



NOTES:

1) ALL RESISTORS ARE IN KΩ 5% EXCEPT ✱ ARE .01%.

2) ALL CAPACITORS ARE IN µF 400V.

3) BYPASS +/- 55V WITH .82µF 100V.

4) $\boxed{n}$ CONNECTOR PIN #.

FIG. 5

NEXT ASSY	PROJECT
APPLICATION	

TOLERANCE UNLESS OTHERWISE NOTED:
DECIMAL: .xx ± .01, .xxx ± .005
FRACTIONAL: ± 1/64
ANGULAR: ± 0° 30'
SURFACE FINISH: 125
BREAK ALL SHARP EDGES AND DEBURR
FINISH

DRAWN: C. Sweeney
ENG'R: R. Sukys
DATE: Nov 24 81
CHECKED
SCALE
MATERIAL

DC EXCITER SCHEMATIC

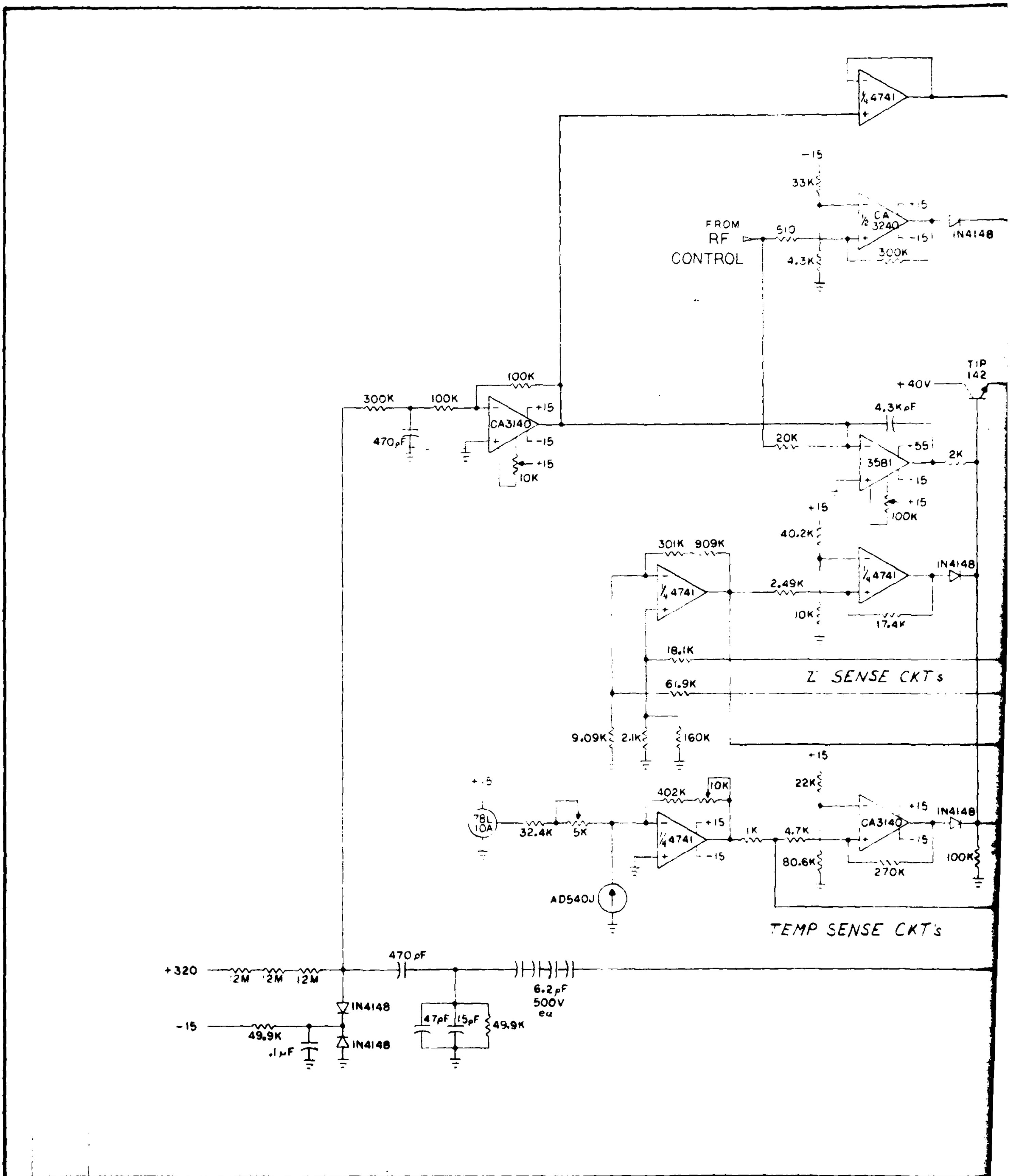
CONTRACT NUMBER

NORTHEASTERN UNIVERSITY

COLLEGE OF ENGINEERING

BOSTON, MASS. 02115

BMS-105R


1/4 4741
-15
33K
1/2 CA 3240
+15
-15
IN4148
FROM
RF
CONTROL
510
4.3K
300K
TIP
142
+40V
100K
300K
100K
CA3140
+15
-15
470pF
10K
4.3KpF
20K
3581
-55
-15
+15
2K
100K
+15
40.2K
301K
909K
1/4 4741
2.49K
10K
17.4K
IN4148
18.1K
Z SENSE CKTs
61.9K
9.09K
2.1K
160K
+15
22K
+15
78L
10A
32.4K
5K
402K
10K
1K
4.7K
80.6K
CA3140
270K
100K
AD540J
TEMP SENSE CKTs
+320
2M
2M
12M
470pF
6.2pF
500V
ea
IN4148
-15
49.9K
.1μF
IN4148
47pF
15pF
49.9K

REVISIONS			
SYMBOL	DESCRIPTION	DATE	APPROVAL

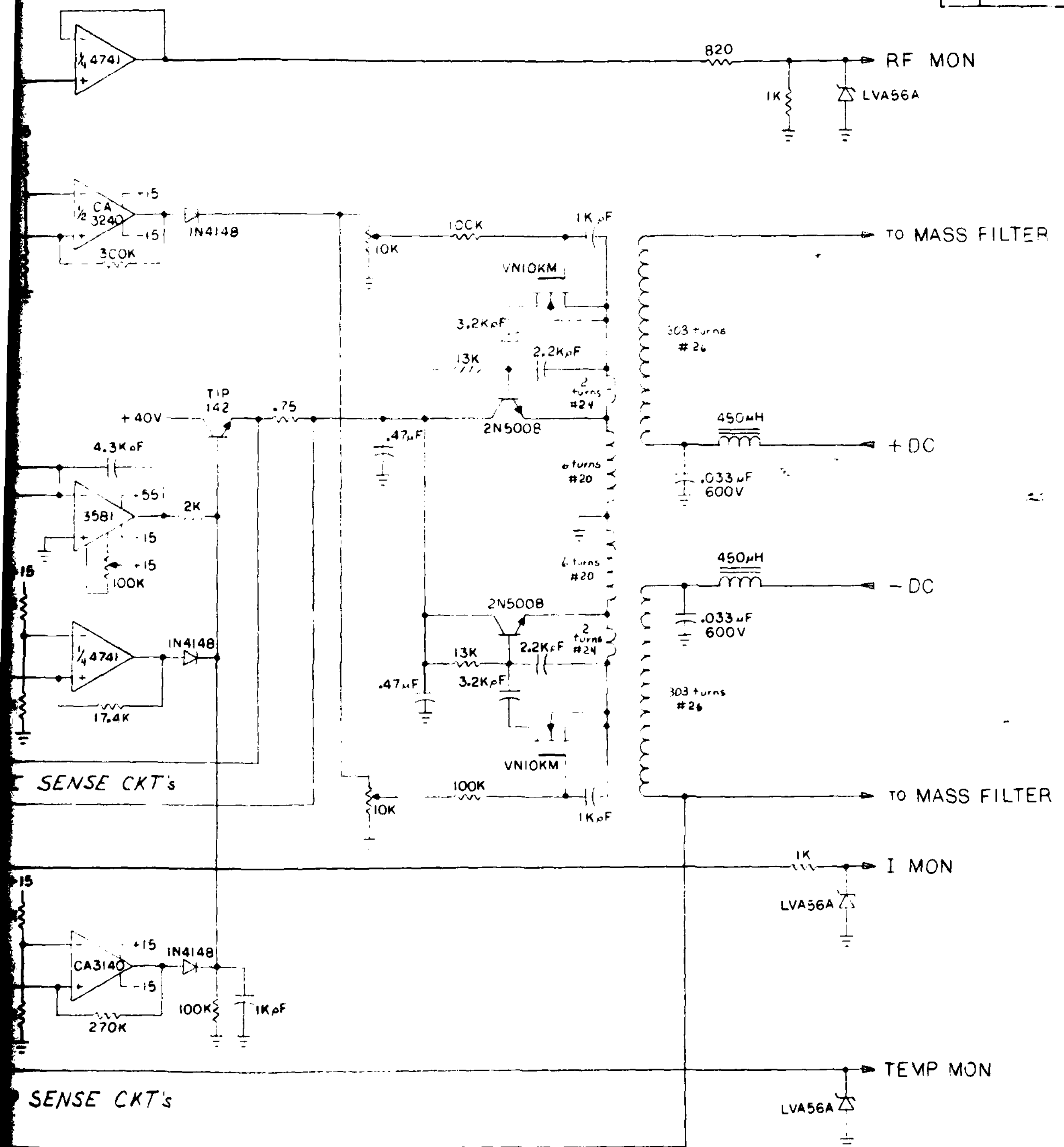


FIG. 6

TOLERANCE UNLESS OTHERWISE NOTED	DRAWN	ENG'R	DATE
DECIMAL: .XX ± .01, .XXX ± .005; FRACTIONAL: ± 1/64; ANGULAR: ± 0° 30'; SURFACE FINISH: 125; BREAK ALL SHARP EDGES AND CORNERS	C. Swaney	T. Whalen	Nov 24 '81
	CHECKED		
	SCALE		
	MATERIAL		

AC EXCITER SCHEMATIC

NEXT ASSY | PROJECT

APPLICATION

FINISH

CONTRACT NUMBER

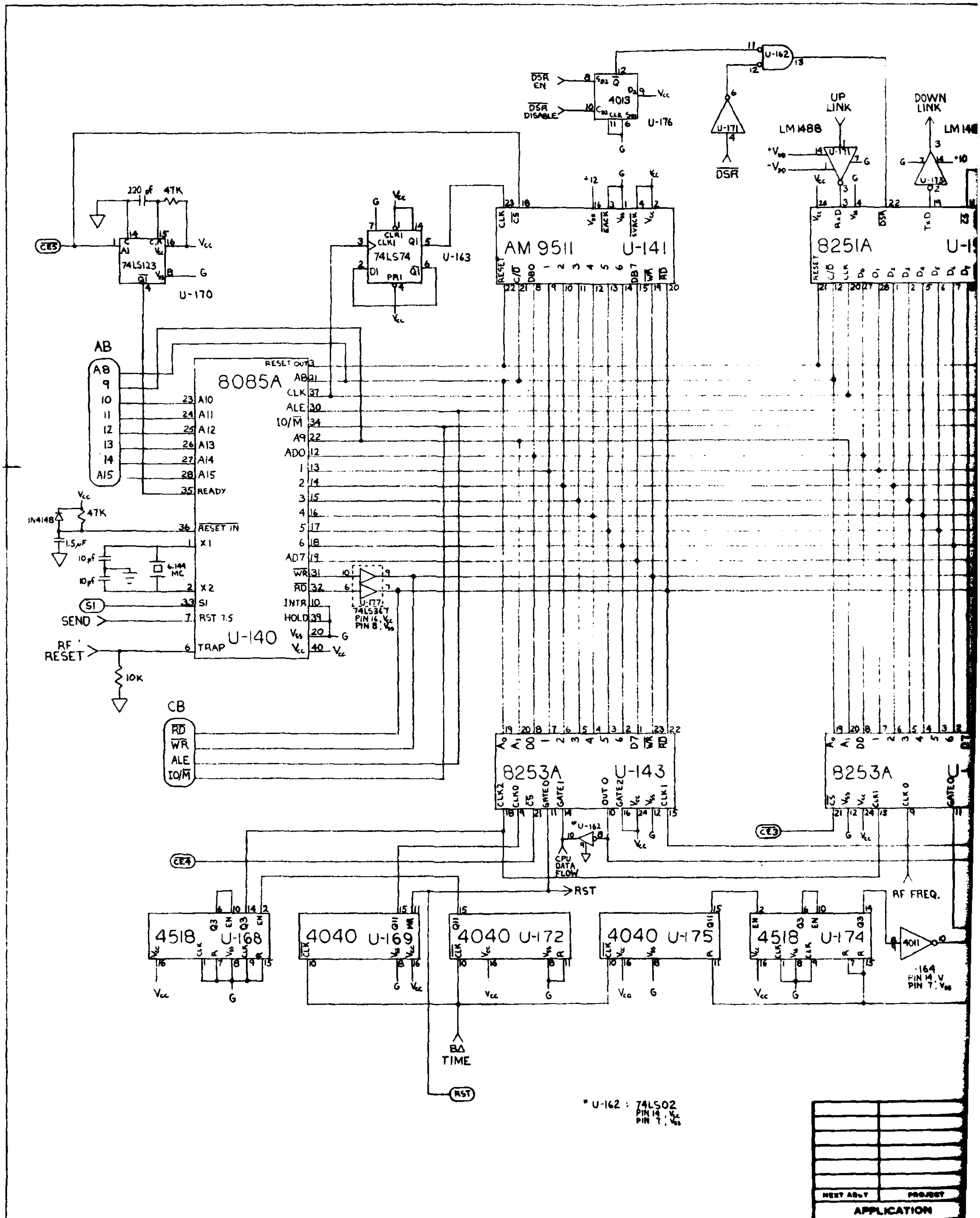
NORTHEASTERN UNIVERSITY

COLLEGE OF ENGINEERING

BOSTON, MASS 02115

BMS-106R

2



8085A
U-140
AM 9511
U-141
8251A
8253A
U-143
74LS123
U-170
74LS74
U-163
4013
U-176
U-162
U-171
LM1488
UP LINK
DOWN LINK
DSR EN
DSR DISABLE
DSR
AB
CB
RD
WR
ALE
IO/M
CE5
CE4
CE3
SEND
RF RESET
S1
74LS367
U-177
PIN 16, Vcc
PIN 8, Vss
CPU DATA FLOW
RST
RF FREQ.
4518
U-168
4040
U-169
U-172
U-175
U-174
4011
BA TIME
* U-162 : 74LS02
PIN 14, Vcc
PIN 7, Vss
NEXT ASSY
PROJECT
APPLICATION

1

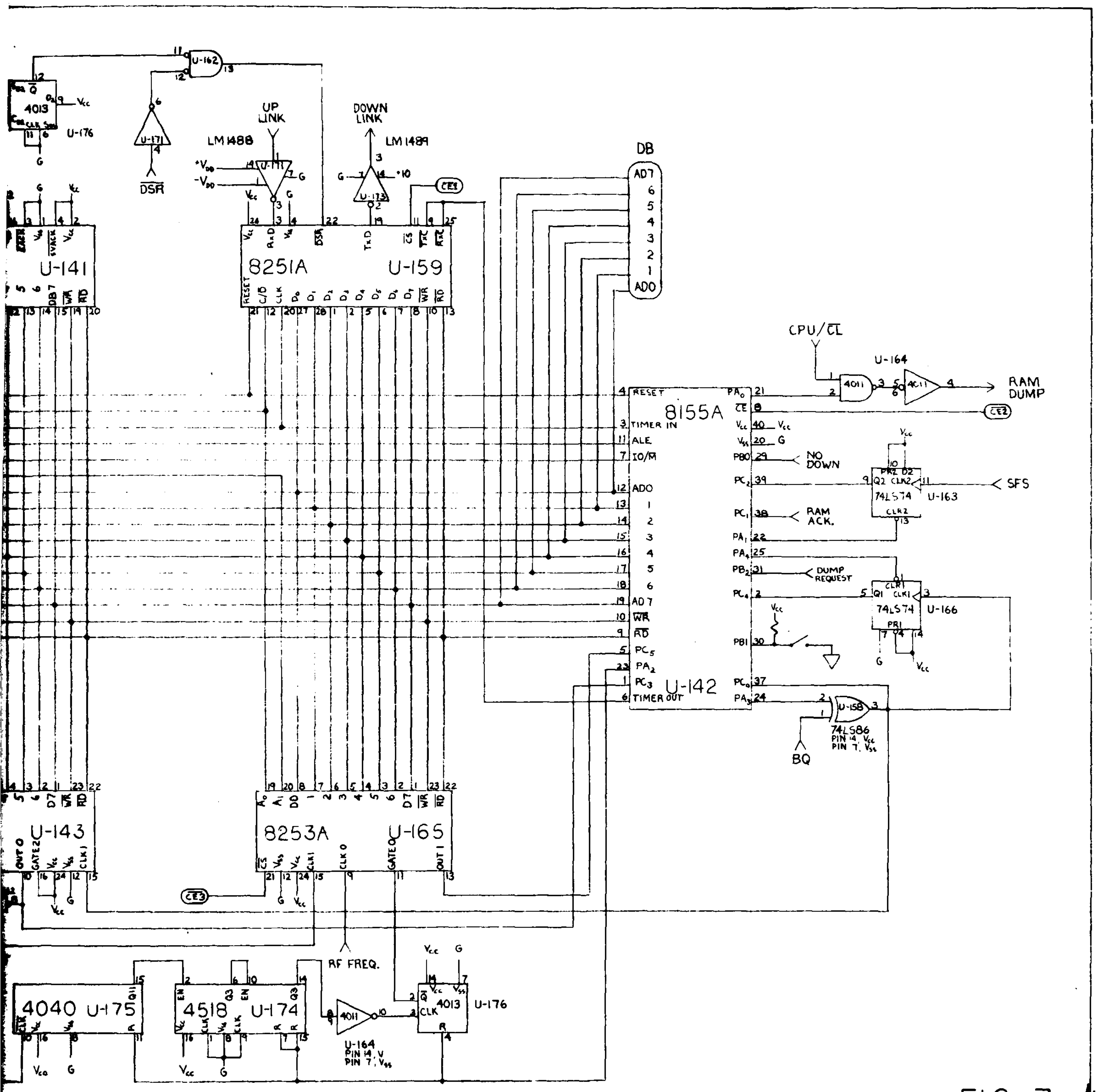


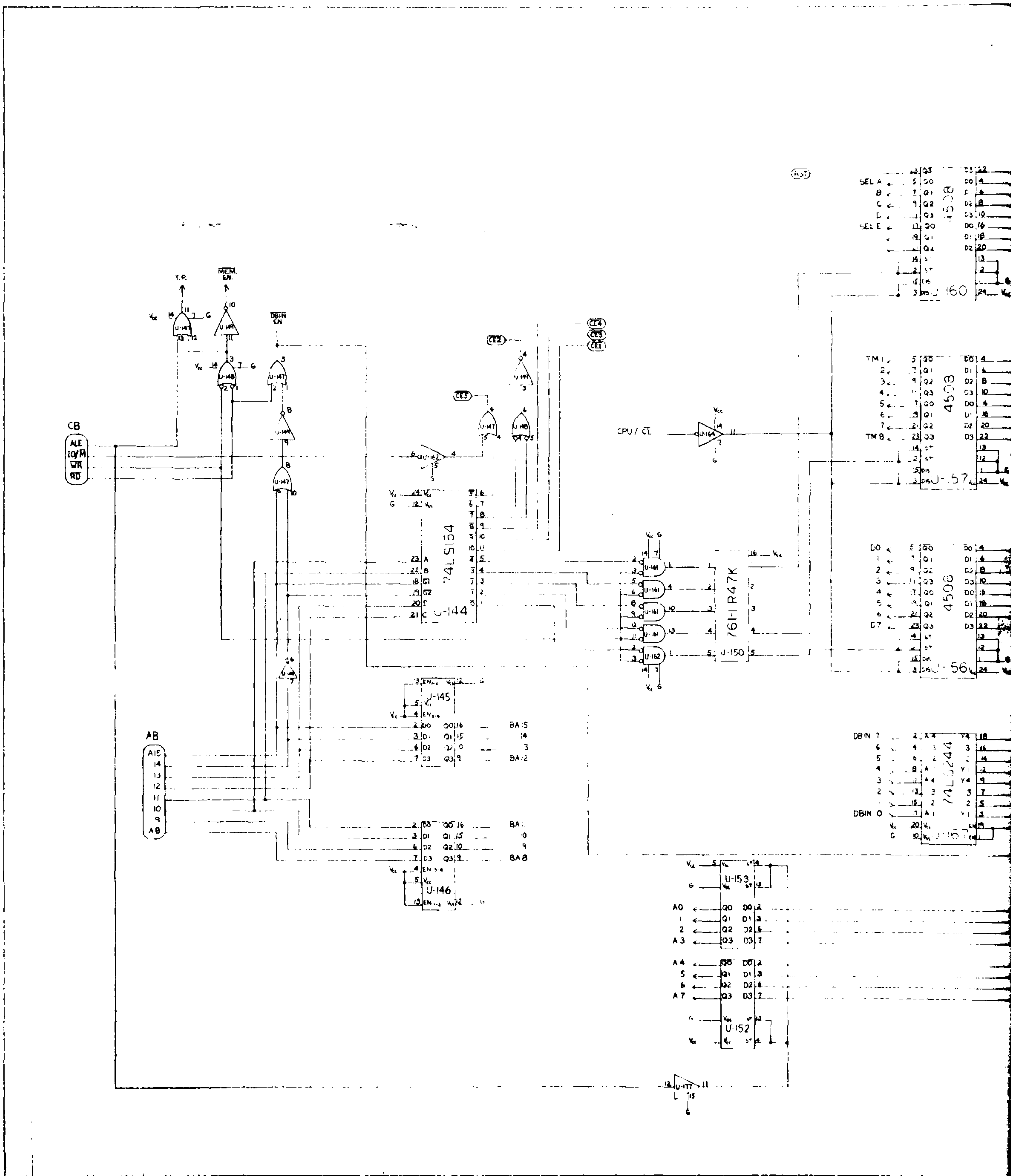
FIG. 7

U-162 : 74LS02
PIN 14, V_{cc}
PIN 7, V_{ss}

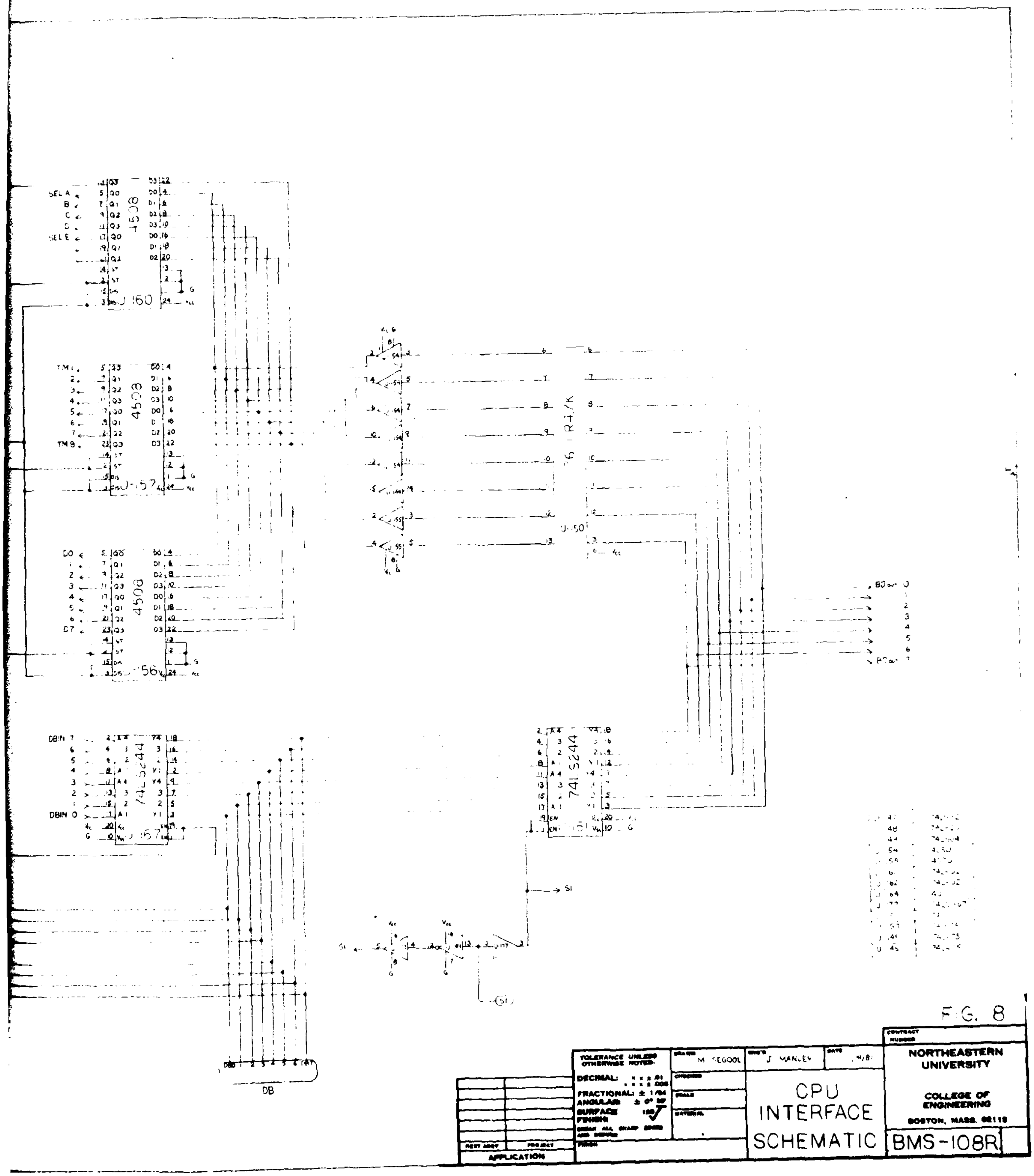
TOLERANCE UNLESS OTHERWISE NOTED:	DRAWN M. SEGOOL	ENG'R J. MANLEY	DATE 11/16/81	CONTRACT NUMBER
DECIMAL: x x ± .01 / x x x ± .005	CHECKED	FLIGHT UNIT CPU SCHEMATIC		NORTHEASTERN UNIVERSITY
FRACTIONAL: ± 1/64	SCALE			COLLEGE OF ENGINEERING
ANGULAR: ± 0° 30'	MATERIAL			BOSTON, MASS. 02115
SURFACE FINISH: 125				BMS-107R
BREAK ALL SHARP EDGES AND BURRS				
FINISH				

NEXT ASSY	PROJECT
APPLICATION	

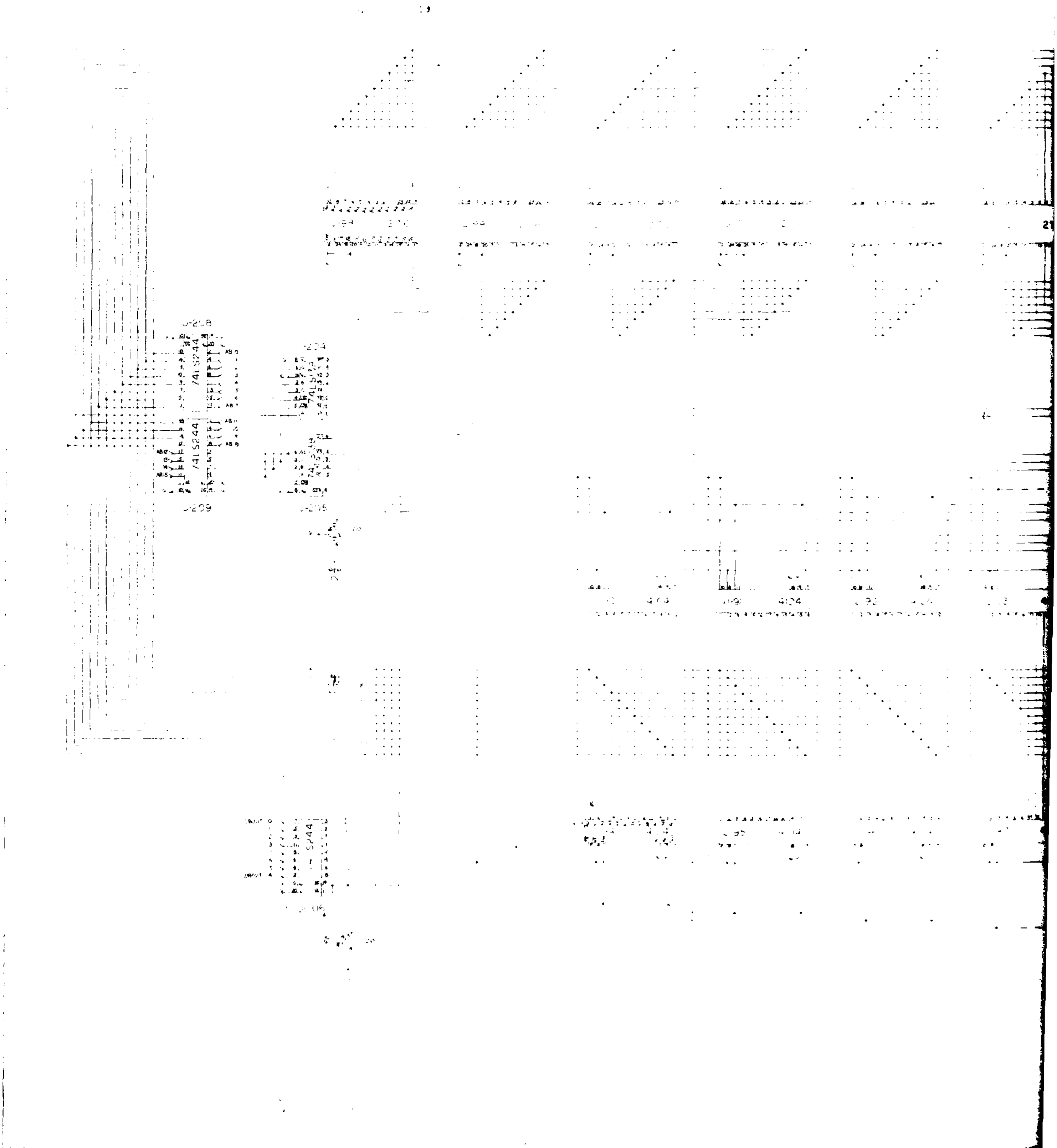
2



T.P.
MEM. EN.
DBIN EN
CB
ALE
IO/M
WR
RD
AB
A15
74LS154
U-144
U-145
U-146
CPU / CE
761-1 R47K
U-150
4508
U-160
U-157
U-156
74LS244
U-153
U-152
SEL A
SEL E
DBIN 7
DBIN 0


4508
U-160
SEL A
SEL E
TM 1
TM 8
U-157
DO
D7
U-156
DBIN 7
DBIN 0
74LS244
U-150
74LS244
U-151
BO out 0
BO out 7
S1
DB
FIG. 8
TOLERANCE UNLESS OTHERWISE NOTED
DECIMAL
FRACTIONAL
ANGULAR
SURFACE FINISH
APPLICATION
M. SEGOOL
J. MANLEY
CPU
INTERFACE
SCHEMATIC
NORTHEASTERN UNIVERSITY
COLLEGE OF ENGINEERING
BOSTON, MASS. 02115
BMS-108R

2


74LS244
74LS244

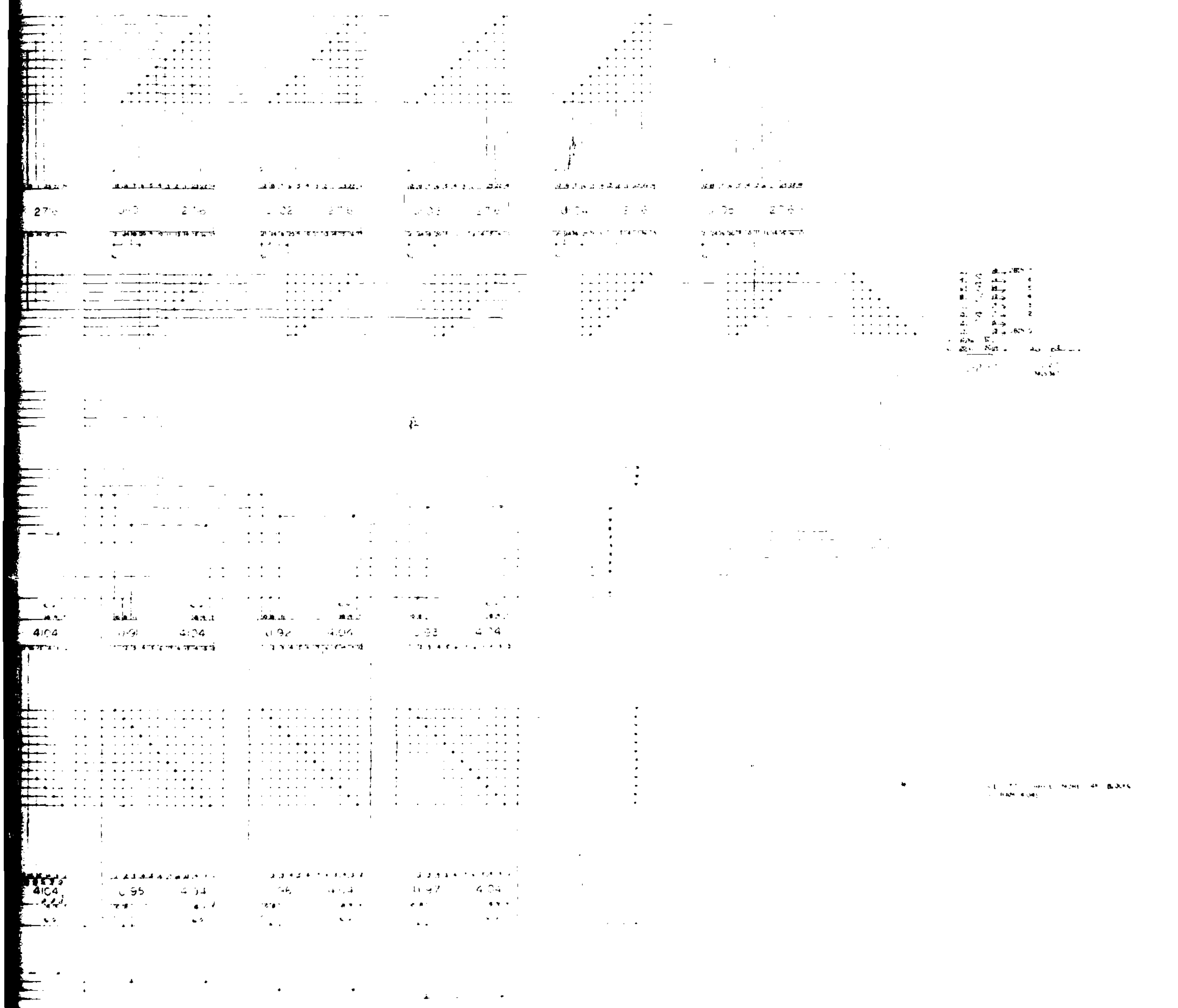


FIG. 9

NORTHEASTERN UNIVERSITY

COLLEGE OF ENGINEERING

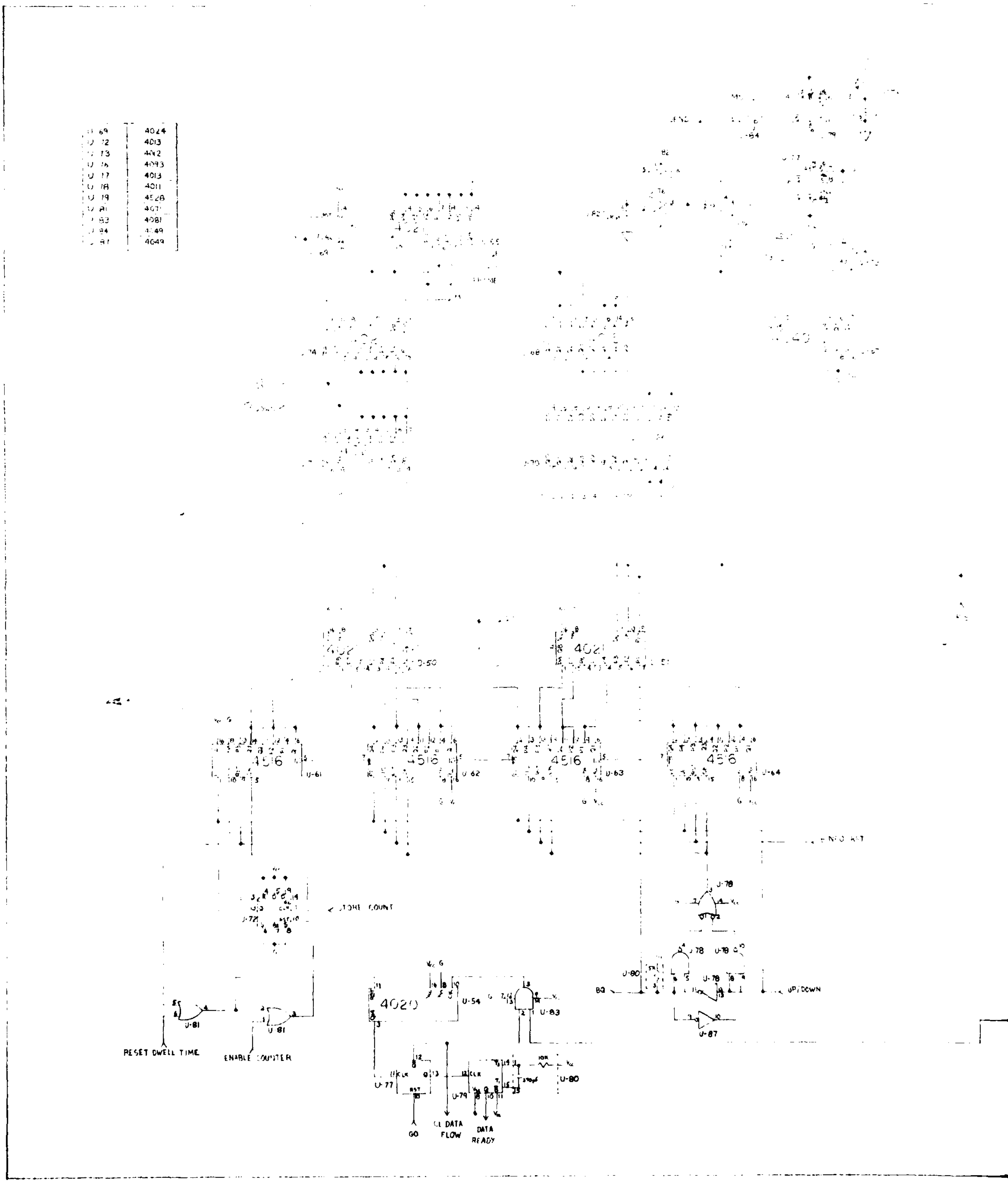
BOSTON, MASS. 02115

MEMORY SCHEMATIC

BMS-109R

APPLICATION

2


U-69 4024
U-72 4013
U-73 4012
U-76 4093
U-77 4013
U-78 4011
U-79 4528
U-81 4071
U-83 4081
U-84 4049
U-87 4049
4516
U-61
U-62
U-63
U-64
STORE COUNT
U-72
UP/DOWN
U-78
U-80
U-87
4020
U-54
U-83
U-81
RESET DWELL TIME
ENABLE COUNTER
U-77
U-79
GO
CL DATA FLOW
DATA READY

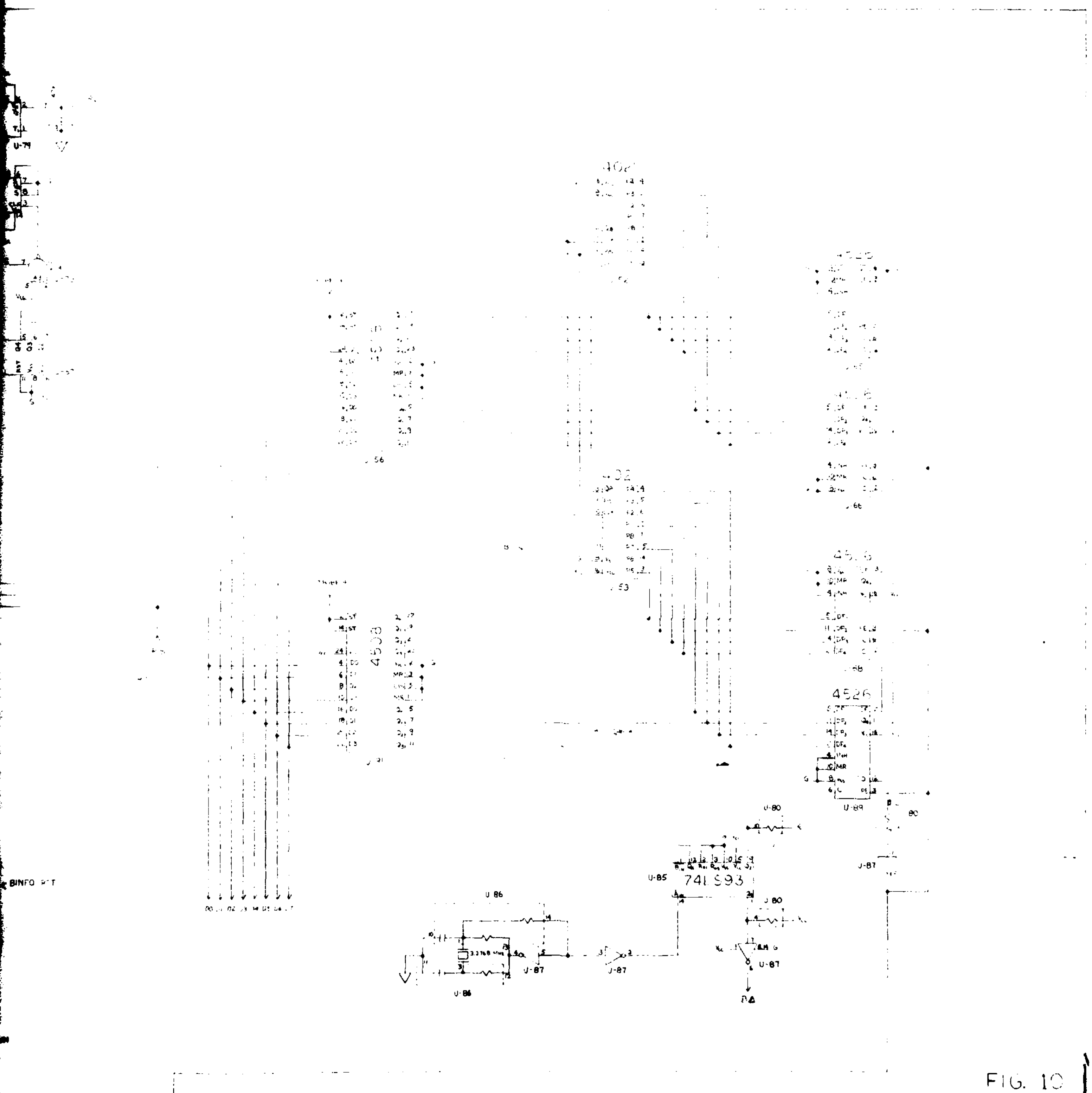


FIG. 10

TOLERANCE UNLESS OTHERWISE NOTED

DECIMAL: ±.01 / ±.005

FRACTIONAL: ± 1/64

ANGULAR: ± 0° 30'

SURFACE FINISH: 125

BREAK ALL SHARP EDGES AND CORNERS

FINISH

DRAWN: M. SEGOOL

CHECKED

SCALE

MATERIAL

ENG'R: J. MANLEY

DATE: /24/81

CL DATA CIRCUITS

CONTRACT NUMBER

NORTHEASTERN UNIVERSITY

COLLEGE OF ENGINEERING

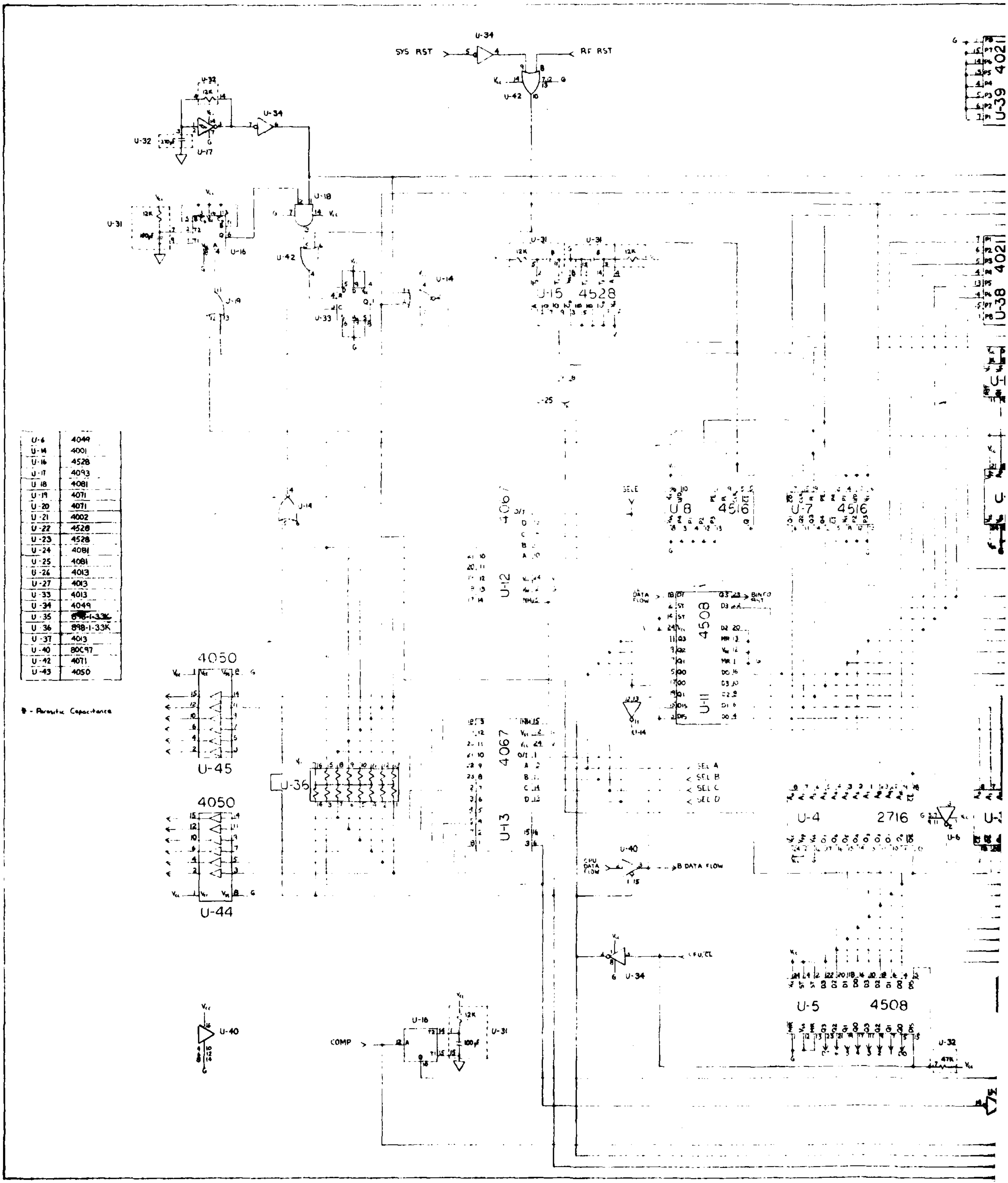
BOSTON, MASS. 02115

BMS-100R

NEXT ASSY | PROJECT

APPLICATION

2



SYS RST
RF RST
U-6 4049
U-14 4001
U-16 4528
U-17 4093
U-18 4081
U-19 4071
U-20 4071
U-21 4002
U-22 4528
U-23 4528
U-24 4081
U-25 4081
U-26 4013
U-27 4013
U-33 4013
U-34 4049
U-35 898-1-33K
U-36 898-1-33K
U-37 4013
U-40 80C97
U-42 4071
U-43 4050
ⓟ - Parasitic Capacitance
4050
U-45
4050
U-44
U-15 4528
U-12
4067
U-13
U-36
U-8 4516
U-7 4516
4508
U-11
U-4 2716
U-5 4508
U-39 4021
U-38 4021
SEL A
SEL B
SEL C
SEL D
CPU DATA FLOW
B DATA FLOW
COMP

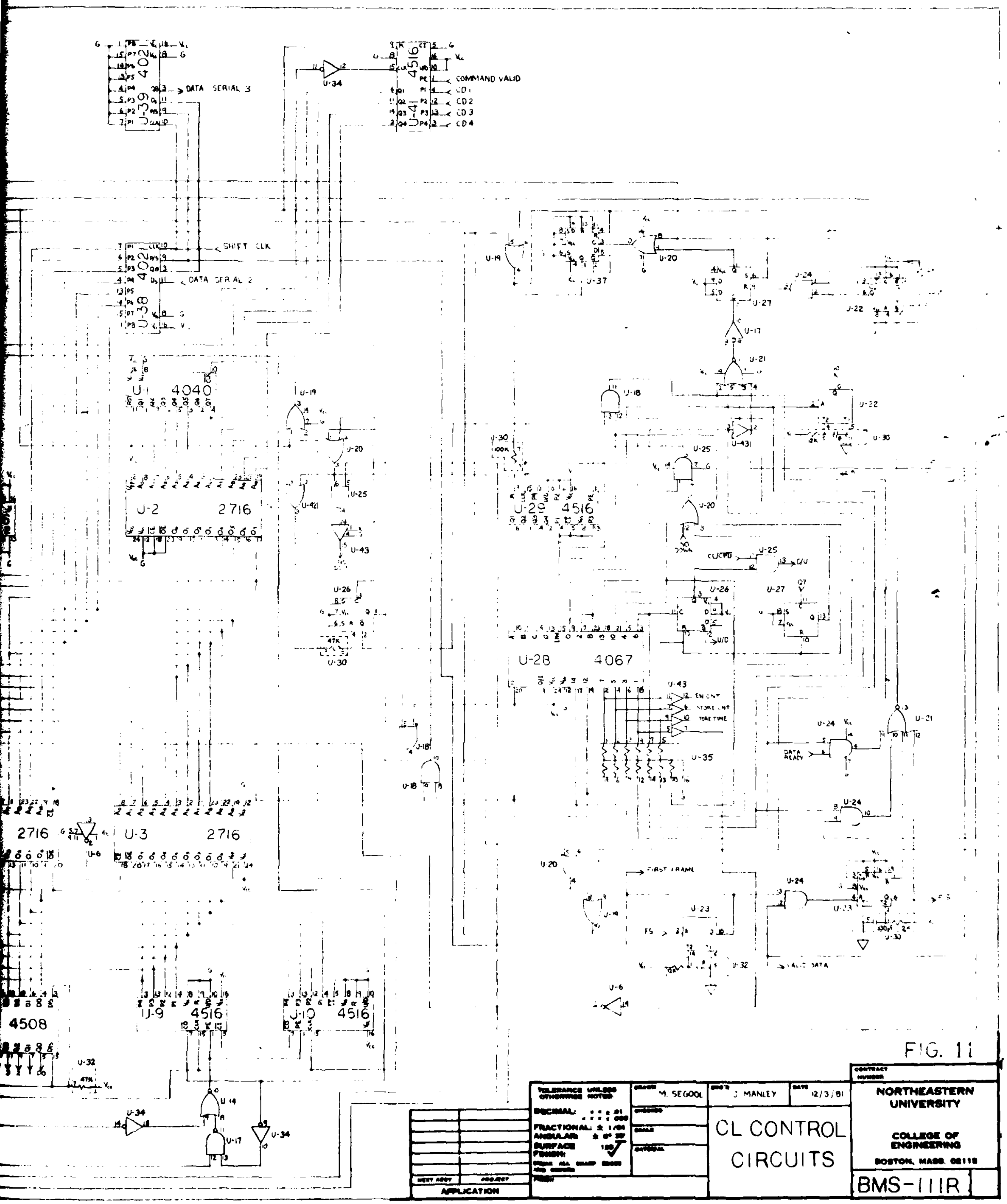
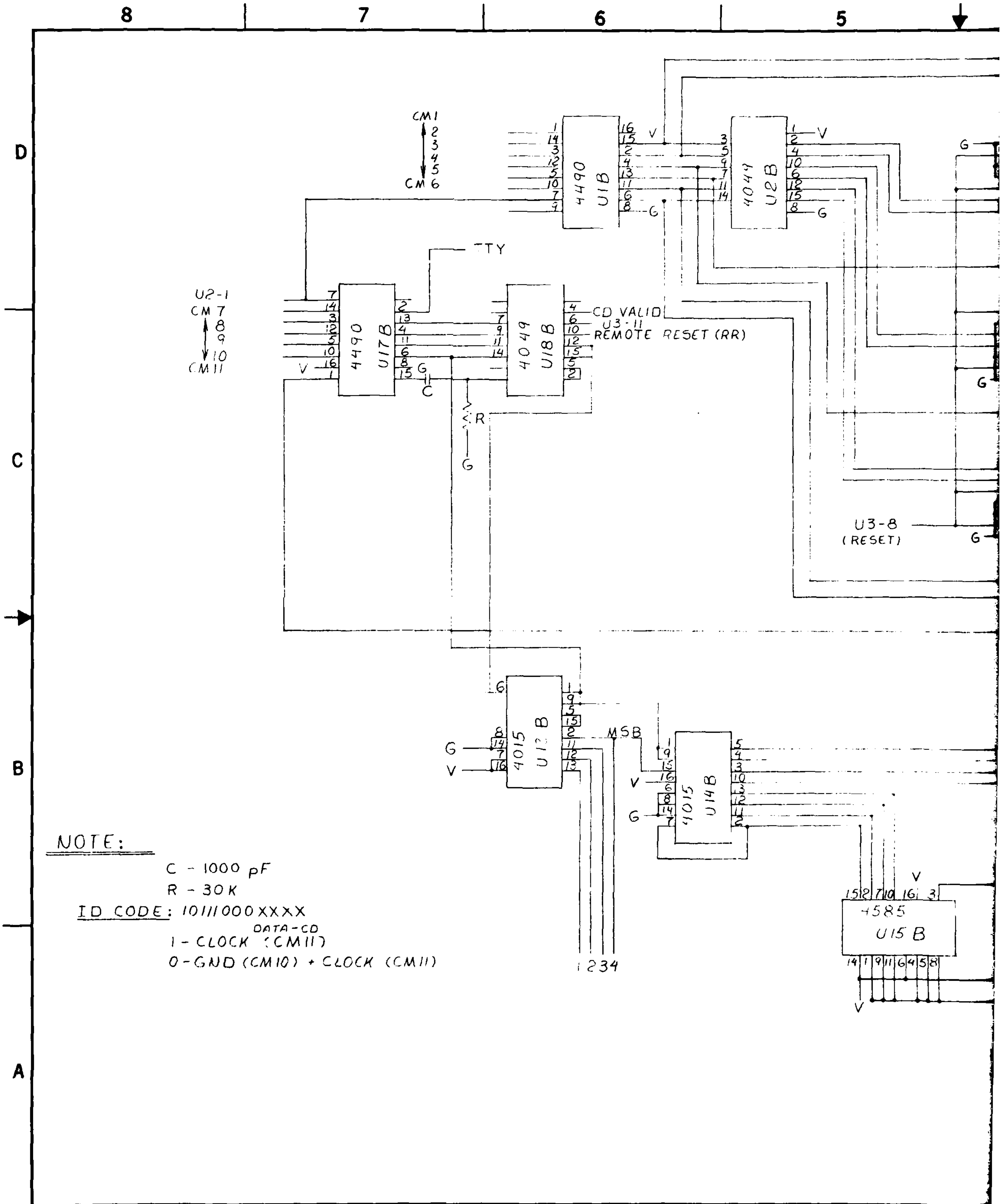

U-39 4021
DATA SERIAL 3
U-41 4516
COMMAND VALID
CD 1
CD 2
CD 3
CD 4
U-34
SHIFT CLK
U-38 4021
DATA SERIAL 2
U-1 4040
U-2 2716
U-3 2716
2716
U-6
4508
U-9 4516
U-10 4516
U-29 4516
U-28 4067
U-37
U-20
U-19
U-27
U-24
U-22
U-17
U-21
U-18
U-25
U-43
U-26
U-30
U-35
U-23
U-32
U-14
EN CNT
STORE CNT
TOKE TIME
DATA READY
FIRST FRAME
VALID DATA
TOLERANCE UNLESS OTHERWISE NOTED
DECIMAL
FRACTIONAL ± 1/64
ANGULAR
SURFACE FINISH
APPLICATION
M. SEGOOL
J. MANLEY
12/3/81
CL CONTROL CIRCUITS
NORTHEASTERN UNIVERSITY
COLLEGE OF ENGINEERING
BOSTON, MASS. 02115
BMS-111R

FIG. 11


8
7
6
5
D
C
B
A
CM1
CM 6
4490
U1B
4049
U2B
TTY
U2-1
CM 7
CM11
4490
U17B
4049
U18B
CD VALID
U3-11
REMOTE RESET (RR)
U3-8
(RESET)
4015
4015
U14B
MSB
4585
U15 B
1 2 3 4
NOTE:
C - 1000 pF
R - 30 K
ID CODE: 10111000XXXX
DATA-CD
1 - CLOCK (CM11)
0 - GND (CM10) + CLOCK (CM11)

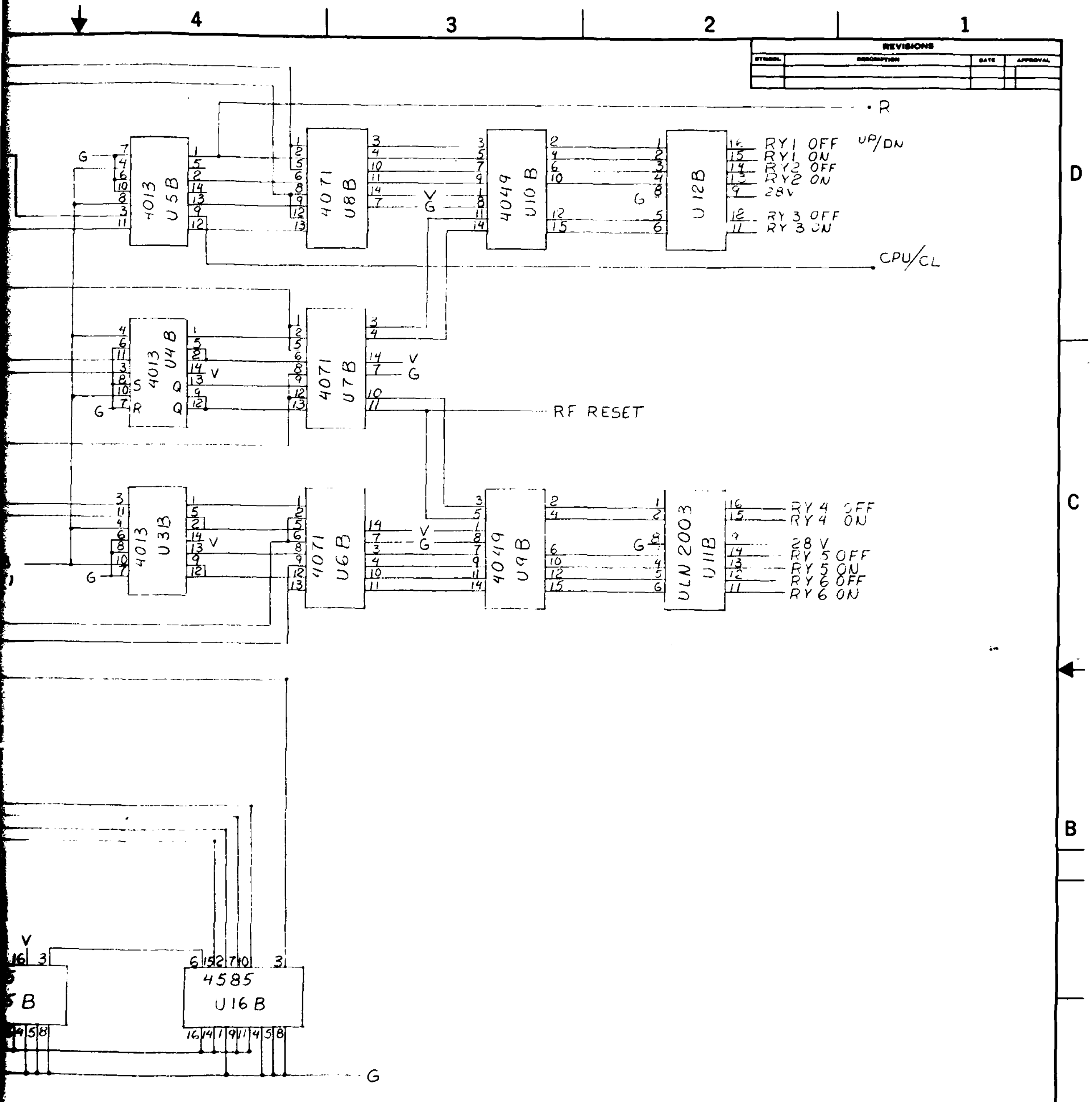


FIG. 12

REVISIONS			
SYMBOL	DESCRIPTION	DATE	APPROVAL

TOLERANCE UNLESS OTHERWISE NOTED:
DECIMAL: .XX ± .01, .XXX ± .005
FRACTIONAL: ± 1/64
ANGULAR: ± 0° 30'
SURFACE FINISH: 125
BREAK ALL SHARP EDGES AND BURRS
FINISH

NEXT ASSY | PROJECT
APPLICATION

DRAWN: S.FARBER
CHECKED
SCALE
MATERIAL
ENG'R: R. Sukys
DATE: 10/16/81

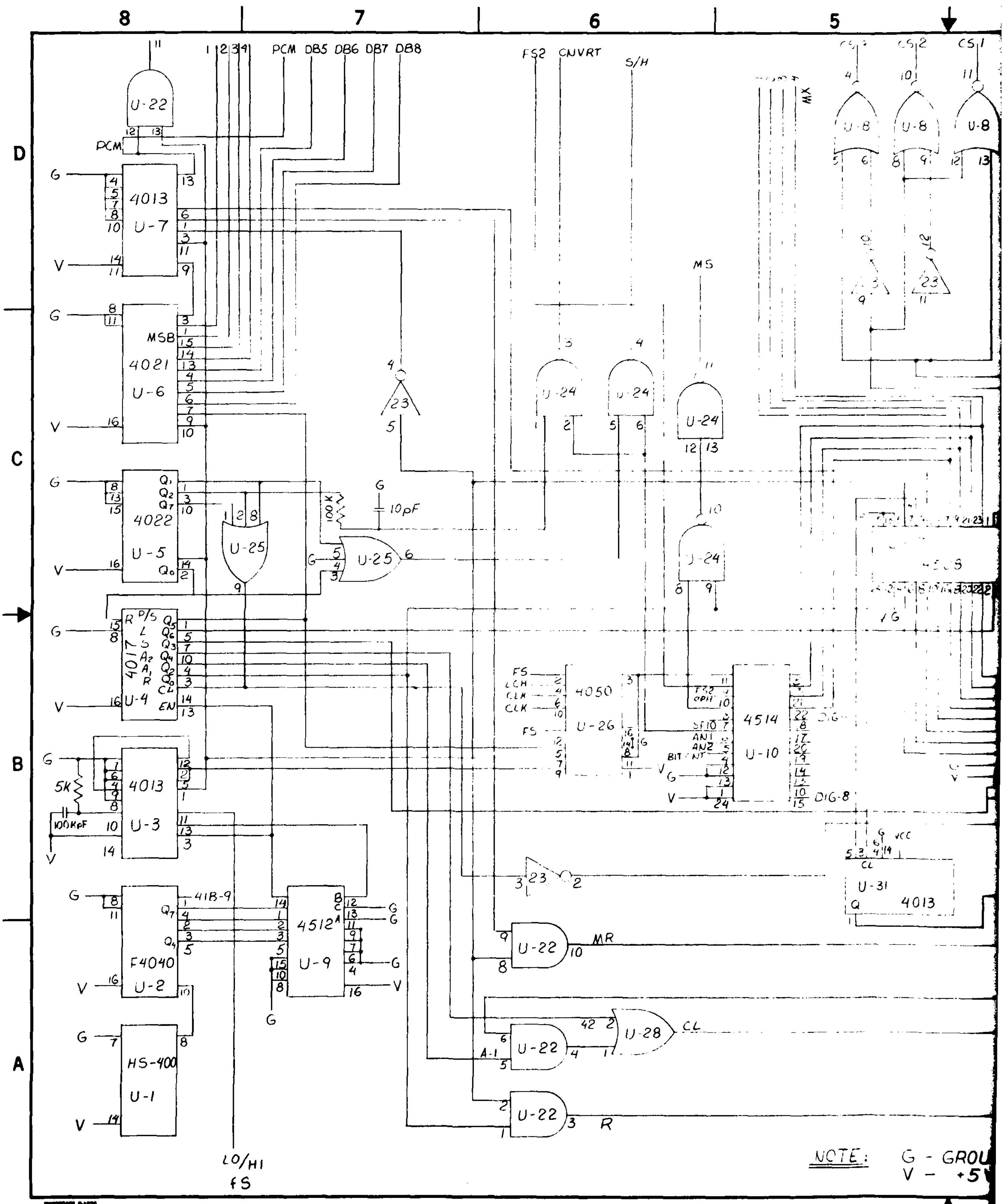
COMMAND/
CONDITIONING
SCHEMATIC

CONTRACT NUMBER

NORTHEASTERN UNIVERSITY
ELECTRONIC RESEARCH LAB
COLLEGE OF ENGINEERING
BOSTON, MASS. 02115

BMS-112R

2


8
7
6
5
D
C
B
A
PCM
DB5
DB6
DB7
DB8
FS2
CNVRT
S/H
MS
CS 3
CS 2
CS 1
MX
U-22
4013
U-7
MSB
4021
U-6
4022
U-5
U-25
10pF
100K
4017
U-4
4013
U-3
5K
100KpF
F4040
U-2
HS-400
U-1
4512
U-9
41B-9
U-23
U-24
U-8
4050
U-26
4514
U-10
DIG-8
U-31
4013
U-28
MR
CL
R
A-1
LO/HI
FS
NOTE: G - GROU
V - +5

1

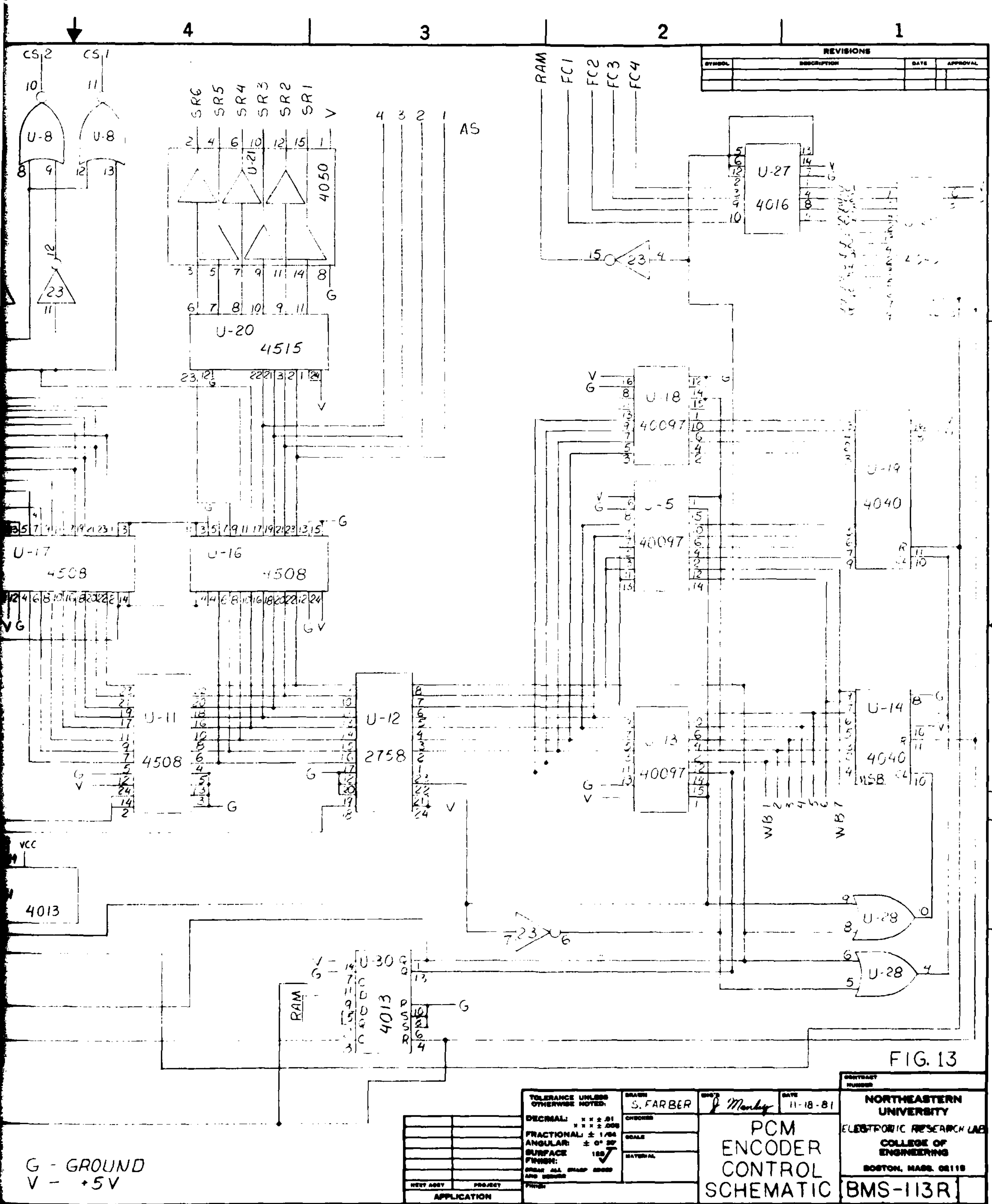
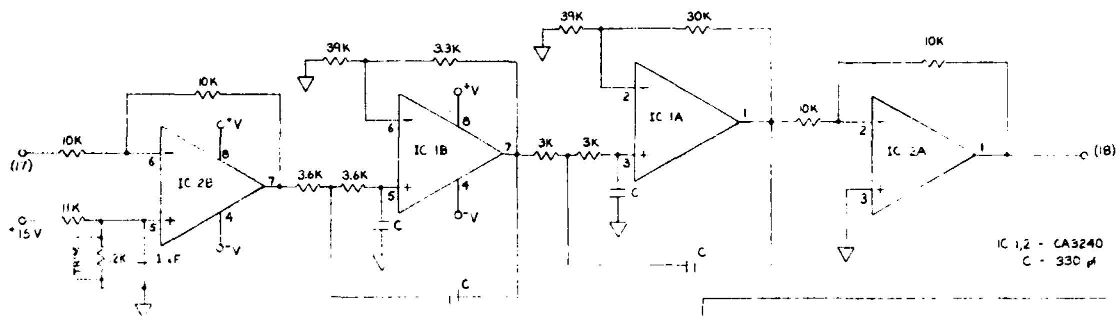
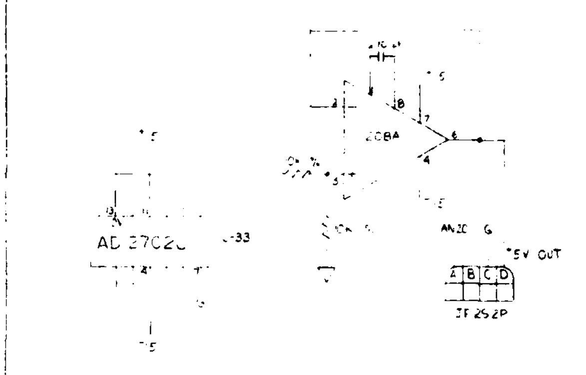

REVISIONS
SYMBOL
DESCRIPTION
DATE
APPROVAL
CS 2
CS 1
U-8
U-8
SR6
SR5
SR4
SR3
SR2
SR1
U-21
4050
4 3 2 1 AS
RAM FC1 FC2 FC3 FC4
U-27
4016
U-20
4515
U-18
40097
U-5
40097
U-19
4040
U-17
4508
U-16
4508
U-11
4508
U-12
2758
U-13
40097
U-14
4040
MSB
WB 1 2 3 4 5 6 WB 7
4013
U-28
U-28
U-30
4013
RAM
G - GROUND
V - +5V
TOLERANCE UNLESS OTHERWISE NOTED:
DECIMAL: x x ± .01 x x x ± .005
FRACTIONAL: ± 1/64
ANGULAR: ± 0° 30'
SURFACE FINISH:
BREAK ALL SHARP EDGES AND BURRS
NEXT ASSY
PROJECT
APPLICATION
DRAWN S. FARBER
CHECKED
SCALE
MATERIAL
FINISH
DATE 11-18-81
PCM
ENCODER
CONTROL
SCHEMATIC
NORTHEASTERN UNIVERSITY
ELECTRONIC RESEARCH LAB
COLLEGE OF ENGINEERING
BOSTON, MASS. 02115
BMS-113R

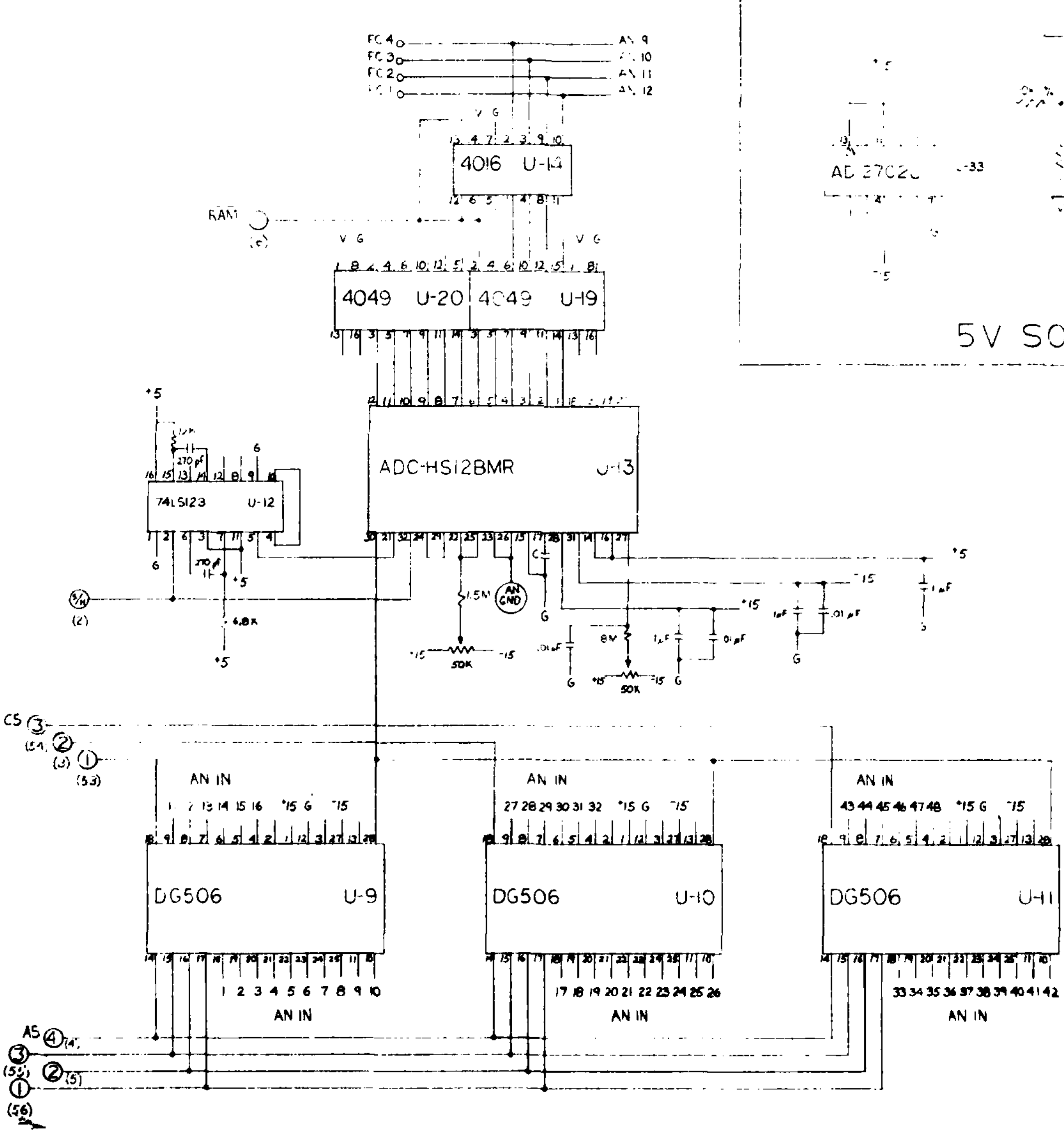
FIG. 13



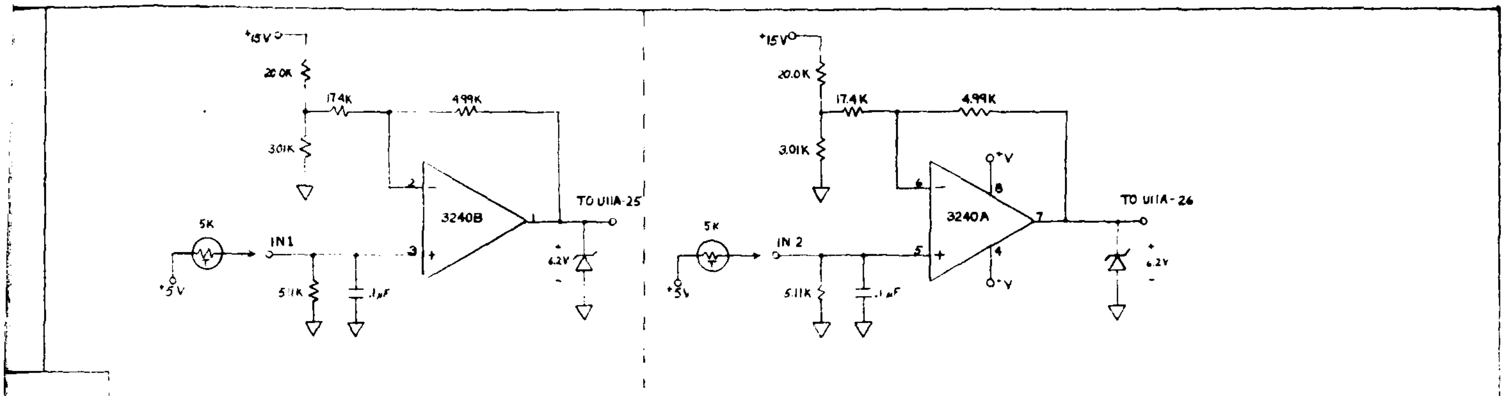
PCM FILTER



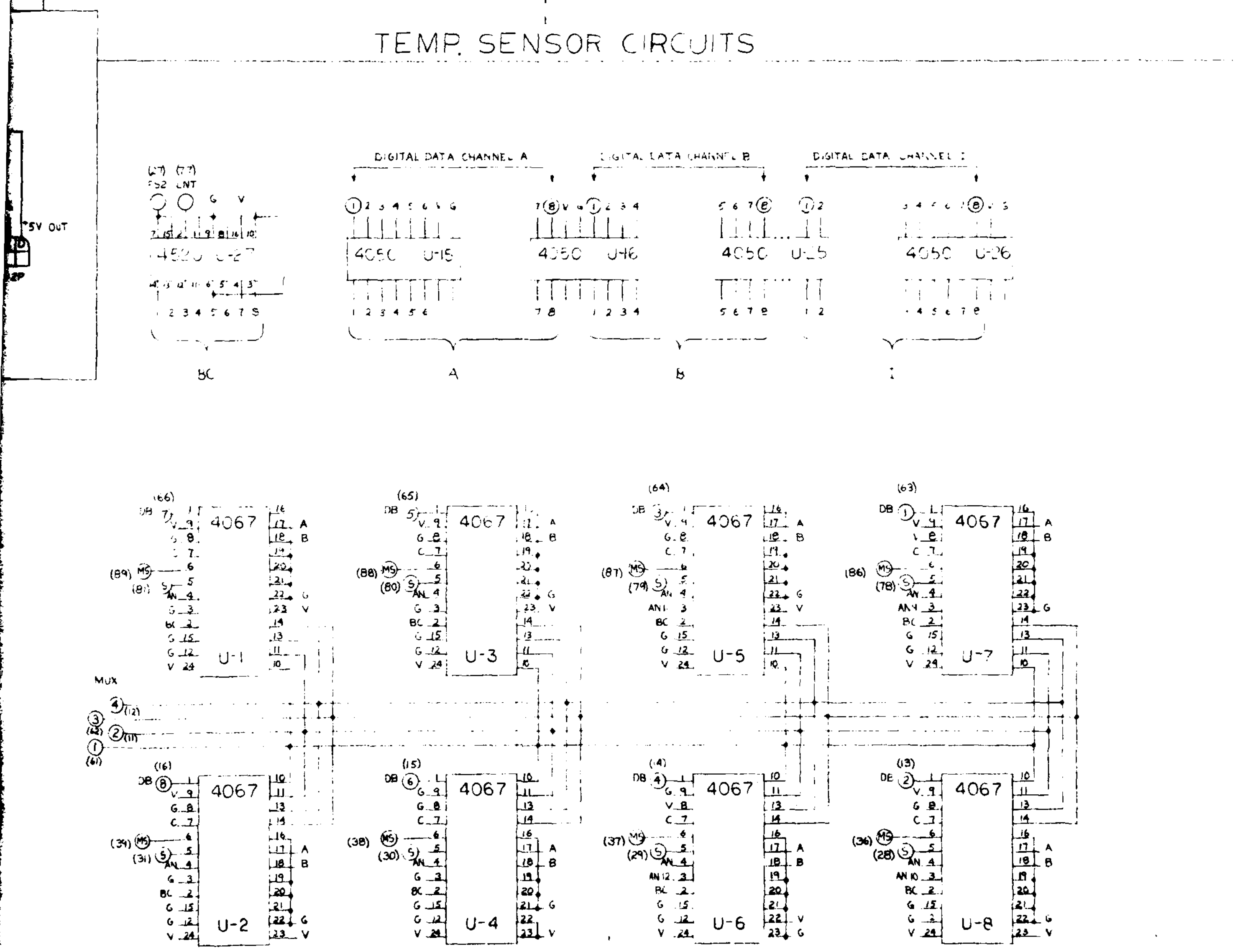
5V SOURCE



ANALOG MUX



TEMP. SENSOR CIRCUITS

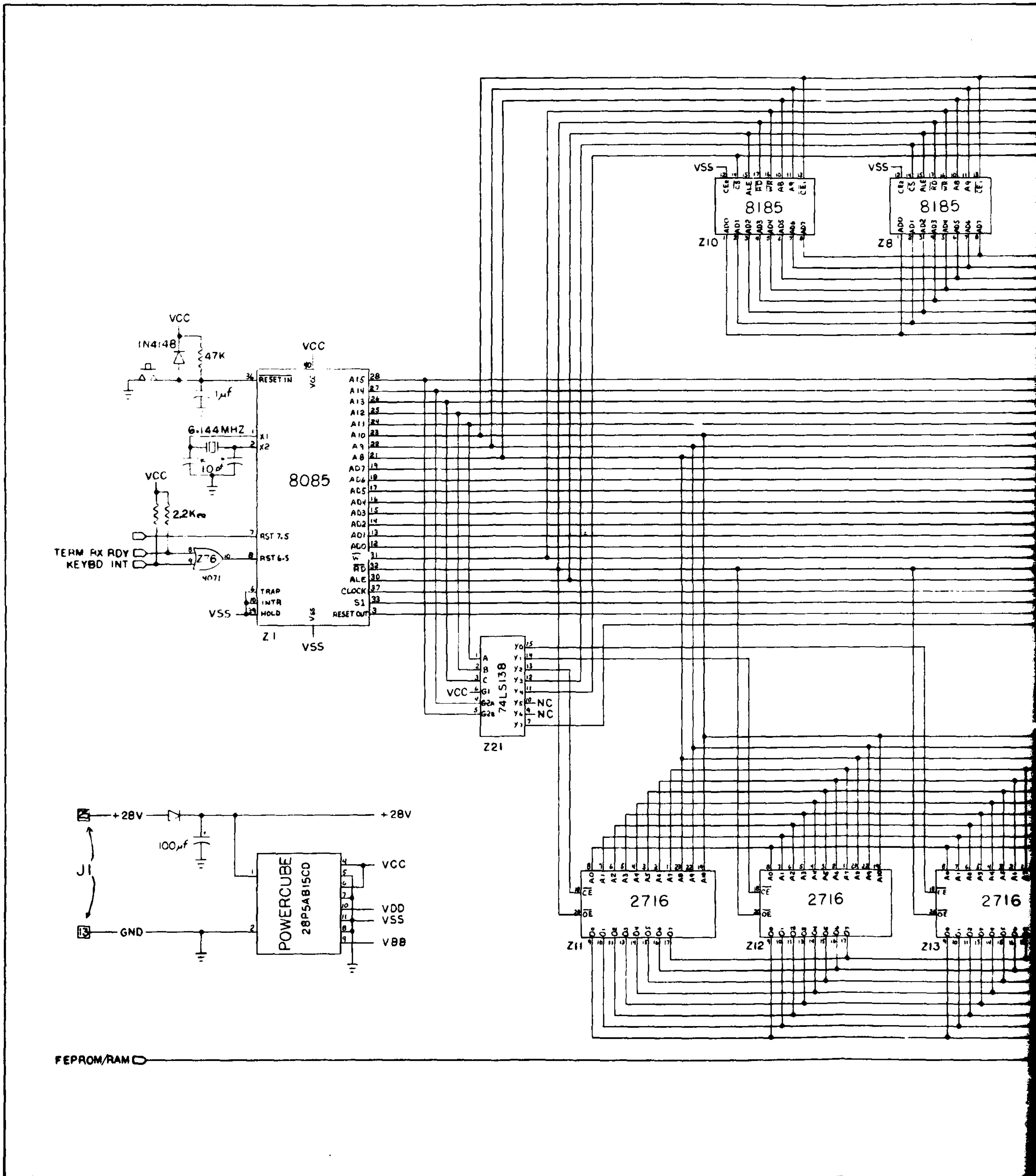


DIGITAL MUX

FIG. 14

TOLERANCE UNLESS OTHERWISE NOTED: DECIMAL, FRACTIONAL, ANGULAR, SURFACE FINISH	DRAWN: M. SEGOOL	ENG'R: R. SUKYS	DATE: 10/9/81	NORTHEASTERN UNIVERSITY
	CHECKED	PCM ENCODER MUX SCHEMATIC		COLLEGE OF ENGINEERING BOSTON, MASS. 02115
NEXT ASSY / PROJECT / APPLICATION	SCALE / MATERIAL			BMS-114R

2



8185
Z10
8185
Z8
VSS
VCC
1N4148
47K
1µf
6.144MHZ
10pf
2.2K
8085
TERM RX RDY
KEYBD INT
Z76
4071
Z1
VSS
74LS138
Z21
NC
+28V
100µf
J1
GND
POWERCUBE
28P5AB15CD
VDD
VBB
2716
Z11
2716
Z12
2716
Z13
FEPROM/RAM

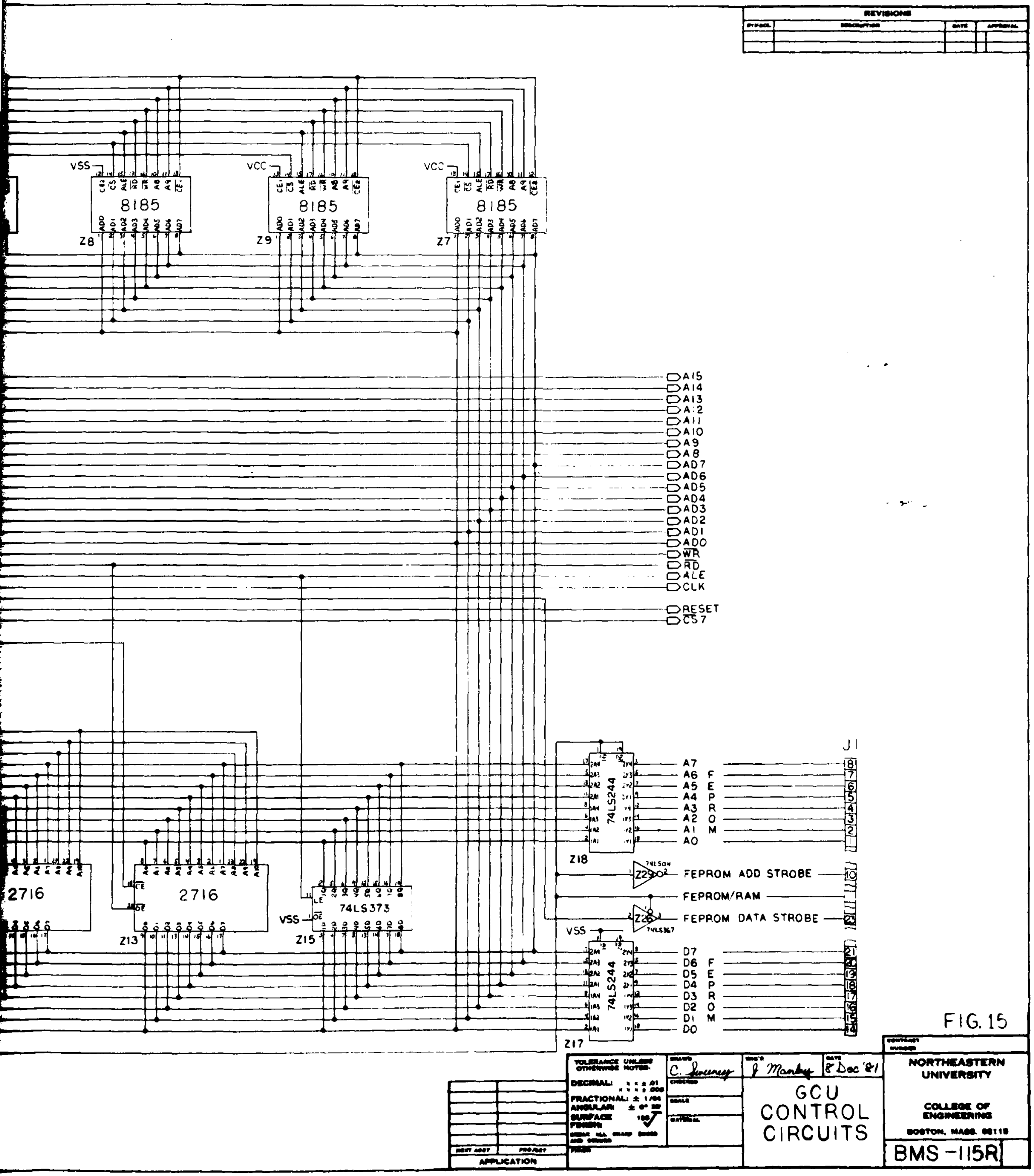
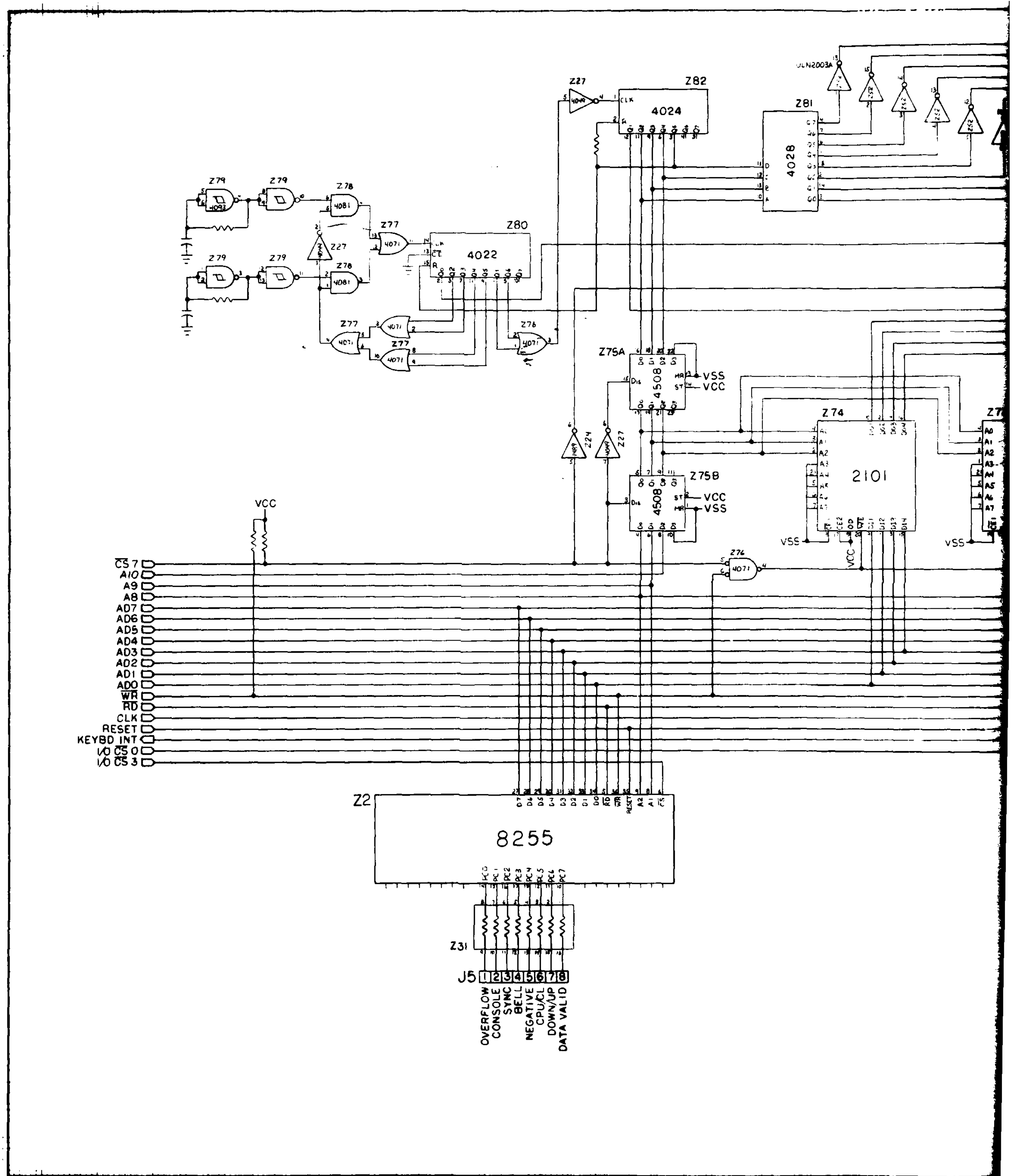

REVISIONS
8185
Z8
Z9
Z7
VSS
VCC
A15
A14
A13
A12
A11
A10
A9
A8
AD7
AD6
AD5
AD4
AD3
AD2
AD1
AD0
WR
RD
ALE
CLK
RESET
CS7
2716
Z13
74LS373
Z15
74LS244
Z18
Z17
J1
FEPROM
FEPROM ADD STROBE
FEPROM/RAM
FEPROM DATA STROBE
74LS04
74LS367
A7
A6
A5
A4
A3
A2
A1
A0
D7
D6
D5
D4
D3
D2
D1
D0
APPLICATION
8 Dec '81
GCU
CONTROL
CIRCUITS
NORTHEASTERN UNIVERSITY
COLLEGE OF ENGINEERING
BOSTON, MASS. 02115
BMS-115R

FIG. 15



Z27
Z82
4024
ULN2003A
Z81
4028
Z79
Z78
4081
4093
Z77
4071
Z80
4022
Z76
Z75A
4508
VSS
VCC
Z74
2101
Z75B
Z24
1489
4049
VCC
CS7
A10
A9
A8
AD7
AD6
AD5
AD4
AD3
AD2
AD1
AD0
WR
RD
CLK
RESET
KEYBD INT
I/O CS 0
I/O CS 3
Z2
8255
Z31
J5
OVERFLOW
CONSOLE
SYNC
BELL
NEGATIVE
CPU/CL
DOWN/UP
DATA VALID

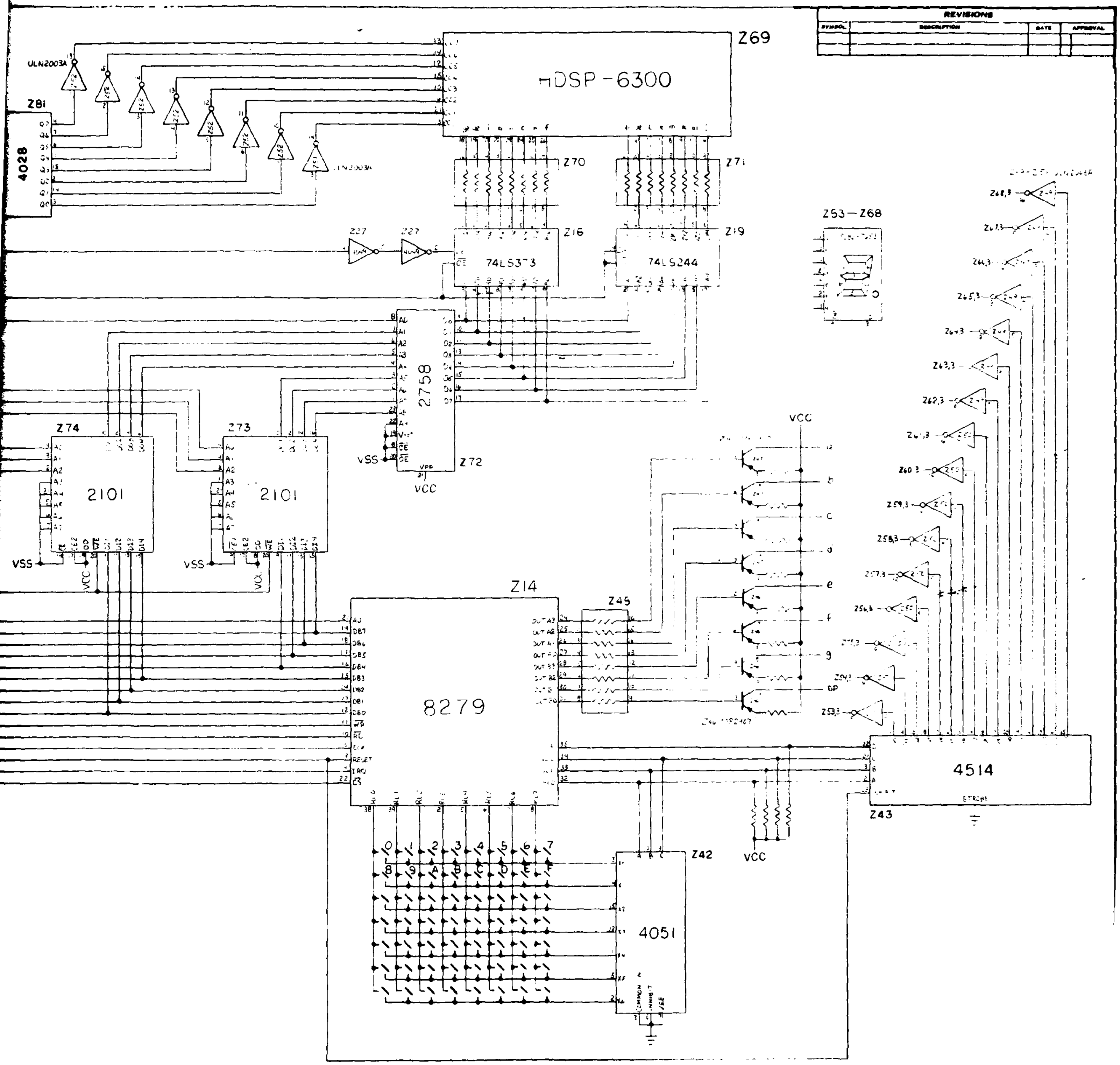


FIG. 17

TOLERANCE UNLESS OTHERWISE NOTED.	DRAWN C. Sweeney	ENG'R J. Manley	DATE 16 Dec 81	NORTHEASTERN UNIVERSITY
DECIMAL: .x ± .01, .xx ± .005	CHECKED	GCU KEYPAD & DISPLAYS SCHEMATIC		COLLEGE OF ENGINEERING
FRACTIONAL: ± 1/64	SCALE			BOSTON, MASS. 02115
ANGULAR: ± 0° 30'	MATERIAL			BMS-117R
SURFACE FINISH				
BREAK ALL SHARP EDGES AND BURRS				
FINISH				

NEXT ASSY	PROJECT
APPLICATION	

III. APPENDIX A - FLIGHT CONTROL PROGRAM

TABLE OF CONTENTS

III. APPENDIX A - FLIGHT CONTROL PROGRAM (continued)

TABLE OF CONTENTS

FLOW CHARTS OF FLIGHT UNIT

SYSTEM FLOW CHART

The flow chart depicts the sequence of events controlled by the balloon borne MPU. Initialization leads into the library cycle. After determining the address of the instruction set to be executed during the next data gathering cycle the MPU leaves the library routine. An inquiry is sent to the ground station about any pending messages. When a message exists the MPU executes the received commands. Next the available memory space is compared with the memory space needed to store the expected data in the upcoming data gathering cycle. If the MPU determines that insufficient space exists, it directs the system into the "RAM DUMP" cycle. The previously stored data is transmitted through the PCM link and, if no contradicting commands are received, the RAM is cleared for the new data. The data gathering cycle is entered where the previously selected instructions from the library or any new instructions received from the ground control unit are executed.

LIBRARY CYCLE

The address of the preselected or commanded repertoire is stored for reference. In absence of any further commands the air-borne control unit will remain within this repertoire indefinitely executing the prearranged program sequence. The repertoire points to the next program where the starting addresses of the instruction sets are listed. Once the address of the next instruction set to be executed is determined and stored the MPU leaves the library cycle.

TTY LINK

First an inquiry is sent to the ground control unit for a message. When a message is available it is repeated 3 times. A match between two out of the three sequences is accepted as an error free transmission. In that case an echo is sent through the PCM to the ground station. A mismatch in all three message sequences initiates a request for a repeat. If the communications link fails to produce an acceptable message the MPU remains in the repeat loop for a limited time before returning to other tasks.

The messages may contain a new instruction set to be executed before the one selected in the library cycle. It also may contain commands to select a new instruction set or a program within any of the repertoires stored in the library. The transmission of the data stored in the RAM or the command to save the contents of the RAM after a transmission are executed before the air-borne control unit returns to other tasks.

DATA CYCLE

The data cycle is entered after the instruction set selected in the library cycle is fetched. Before transmission to the DAC's the selected amu control code word is corrected to compensate for the RF frequency drift. The correction process may be disabled by a switch in the flight control unit. When the appropriate control codes are latched into their respective DAC's the data gathering interval is entered. It's determined by the dwell time instruction. The minimum dwell time = 10ms and may be incremented in 5ms steps. Once the data is gathered at the given amu level it may be corrected for noise induced errors. The adjustment may be waved by a ground command or automatically by passed if the ion count exceeds 800H.

In the "ACCUMULATE" mode the mass filter repeatedly scans over a range of amu domains. The count obtained at each of the scan increments is accumulated in the RAM.

When in the bias switching mode the biases may be switched or swept while the amu control signals are kept at a selected level.

When finally the data gathering process ends and the real time data has been transmitted the data stored in the RAM is converted into counts per second.

INT 7.5

The request for this interrupt and the subsequent subroutine originates in the PCM system. The routine feeds the MS and the RAM data to the PCM encoder. Upon request the MPU places a data byte stored in the PCM buffer onto the MS-to-PCM bus. It also determines the sequence in which the various MS data bytes are transmitted. Data transmission from the RAM is also controlled by this routine.

FREQC.

The subroutine determines the necessary correction factor to stabilize the operation of the MS. It calculates the multiplier by which the amplitude of the quadrupole excitation signal must be modified to compensate for the frequency drift of the RF oscillator. The multiplier $M=(f/f_O)^2$, where f_o is the nominal frequency and f is the actual frequency of the RF oscillator. Thus $V=MV_o$ where V is the corrected amplitude and Vo is the nominal amplitude for a given AMU. A switch on the CPU BOARD disables this operation if desired.

STORE

Stores data into the RAM. Points to a new location for data storage.

TRANSFR

Transfers the control data to an appropriate MS control part.

BIASOT

Transfers the 5 bias control codes to their respective DAC's

RECALL

Retrieves the information stored in the RAM for transmission through PCM. Points to the next data byte.

MAIN

Finds the next instruction set to be executed.

DATCOL

Executes the waiting loop during the MS data collection interval. The dwell time at a given amu step determines the waiting period.

WAIT

Executes the waiting loop during the calculation carried out by the arithmetic unit (AM9511).

NEG

Informs the arithmetic unit that that data in TOS (Top of Stack in the arithmetic unit) is negative.

STZERO

This routine adjusts the 8253 counter input for a proper reception of the first data bit from the FF in the MS, which does not have a reset line leading to it. It also resets the first negative transition detector used in conjunction with "CORDAT" subroutine.

ENDING

Ends the execution of an instruction set by converting the data collected and stored in the RAM into counts per second. It also indicates the end of real time data transmission for that interval by clearing the PCM frame.

ADDDAT

Adds newly collected data to the previously collected data when in the "ACCUMULATE" mode.

BIAS

This subroutine performs bias sweep operation with the AMU sweep kept constant at a selected level. This operation may also be performed in the "ACCUMULATE" mode. For that purpose the starting location of the RAM block assigned to store the data is noted by the "SAVE TM END" operation.

CORDAT

Corrects the data count stored in the 8253 counter to the actual count obtained during the specified dwell time. The incoming data count is first divided by 2 in the MS. Also, the 8253 does not respond to the first negative going input transition. The "STZERO" subroutine assures that the input to the 8253 is at ZERO before the count starts and also resets the first negative transition detector. "CORDAT" uses these conditions to calculate the actual ion count. When 8253 shows a ZERO count and the transition detector is RESET, when the count is either ZERO or ONE depending on the status of the data line.

When the transition detector is SET then the count is 2 or 3 again depending in the status of the data line. When the 8253 count is other than ZERO than the count is multiplied by 2 and 2 or 3 are added depending on the status of the data line.

AMU

Controls the AMU scan defined by an instruction set. Performs correction on the amu control word to compensate for RF frequency drift when in that mode. Also operates in the "ACCUMULATE" mode.

COLECT

Collects the ion count. Checks if correction for noise induced errors is warranted. Performs the corrections when needed. Store the data with the proper amu identification and the instruction set in the RAM for delayed transmission.

RUN

Fetches the instruction set to be executed and stores it in the RAM. The subroutine also stores the starting address of the instruction set to be used as program identification during the data transmission from the RAM. The available memory space (in the RAM) to store the data from a pending execution of an instruction set is calculated. When insufficient space exsists the "RAM DUMP" process is executed. When the expected data exceeds the total capacity of the RAM, the instruction set is disregarded. If the memory space is sufficient to accommodate the data a,24 bit synch word is stored in the RAM followed by the 22 byte instruction set and a 2 byte program ID. Then the control words are sent to the appropriate DAC's and the data gathering begins.

DUMP

Controls the transmission of data from the RAM to the PCM system.

PACK

Combines two bytes into one 16 bit word.

CNVRT.

Converts a memory block of adjacent ASCII characters into the binary code acceptable to the system. The code is stored into the same block of memory.

RECEV.

Receives one data byte from the GCU. Checks the time allocated for the communication with the GCU. Abandons the communications attempt when the time linit is reached.

TRNSMT.

Determines the start of a new PCM frame, updates the TTY buffer location, sets the "TTY ACTIVE" flag for the status word and transfers the TTY data into the communications downlink.

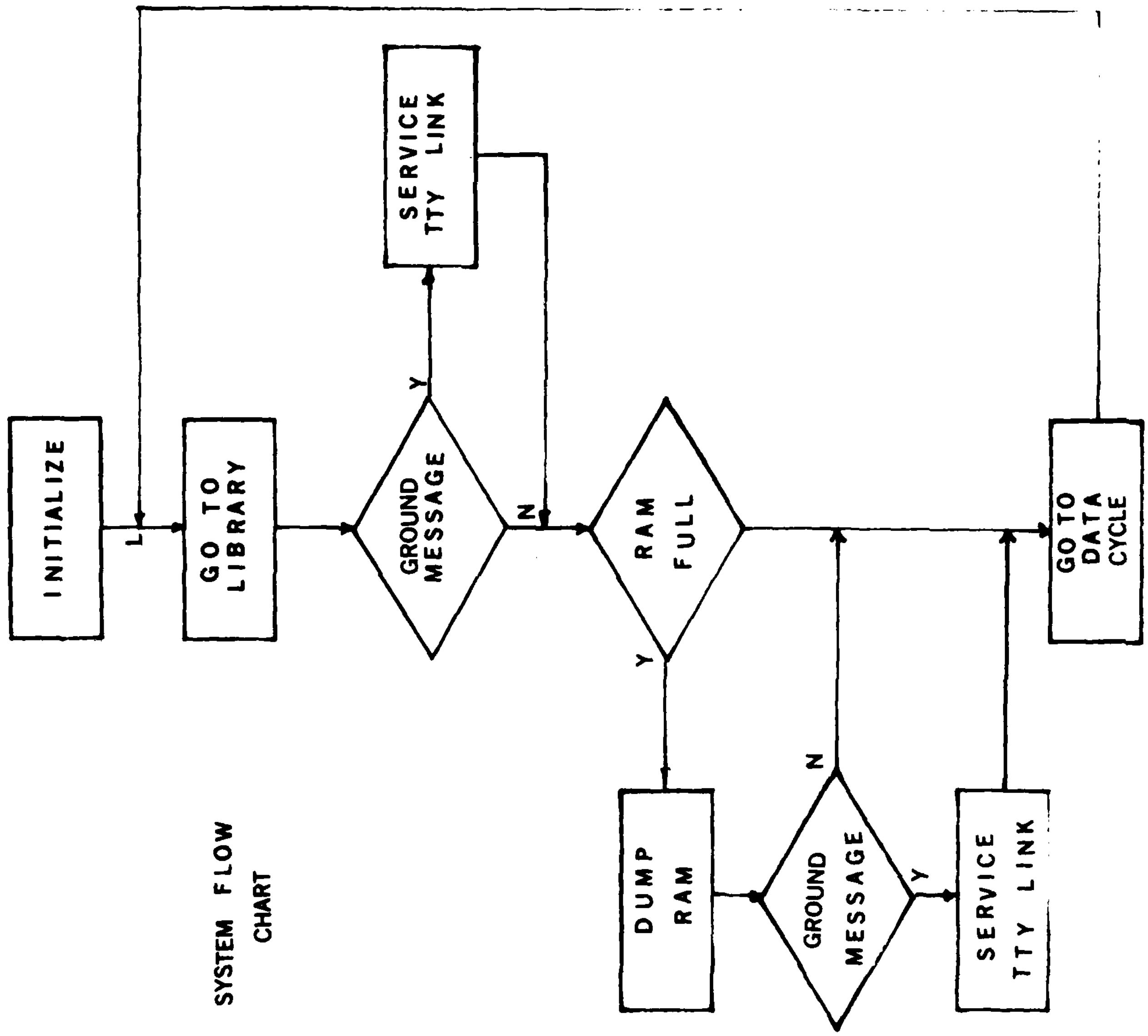
CMPDH

Compares registers HL and DE.

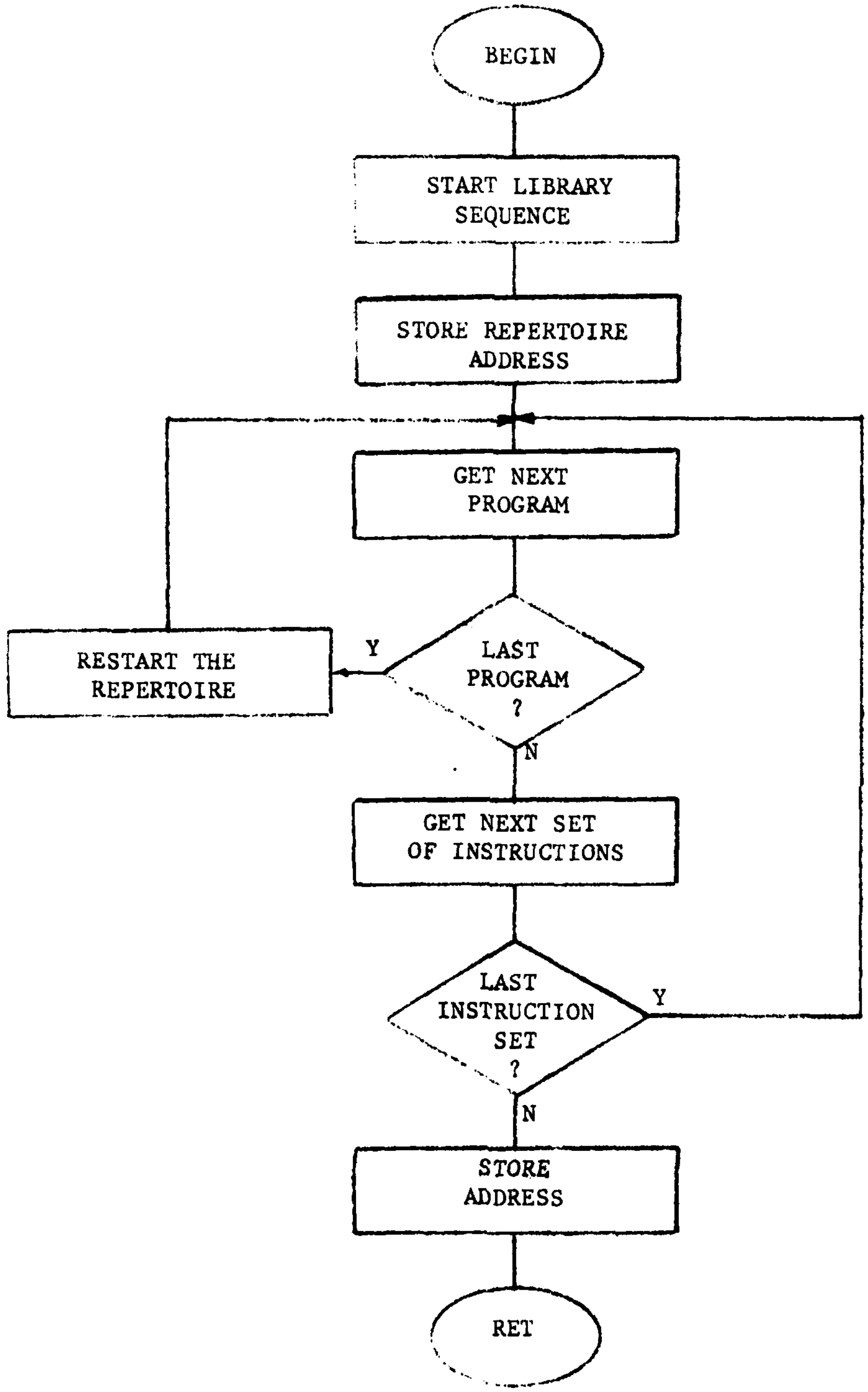
If HL >DE CY=1

HL ≤DE CY=0

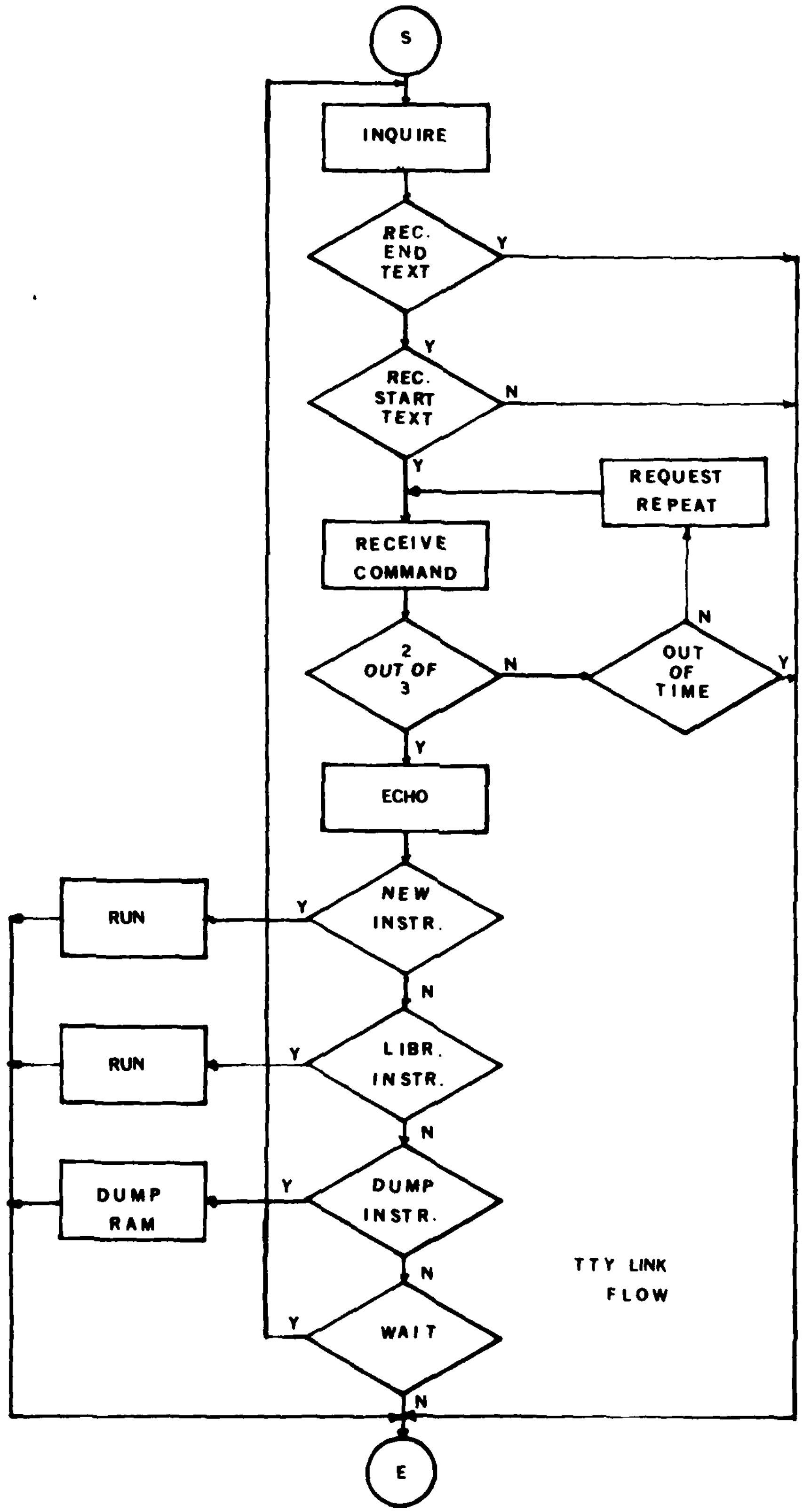
HL =DE Z=1

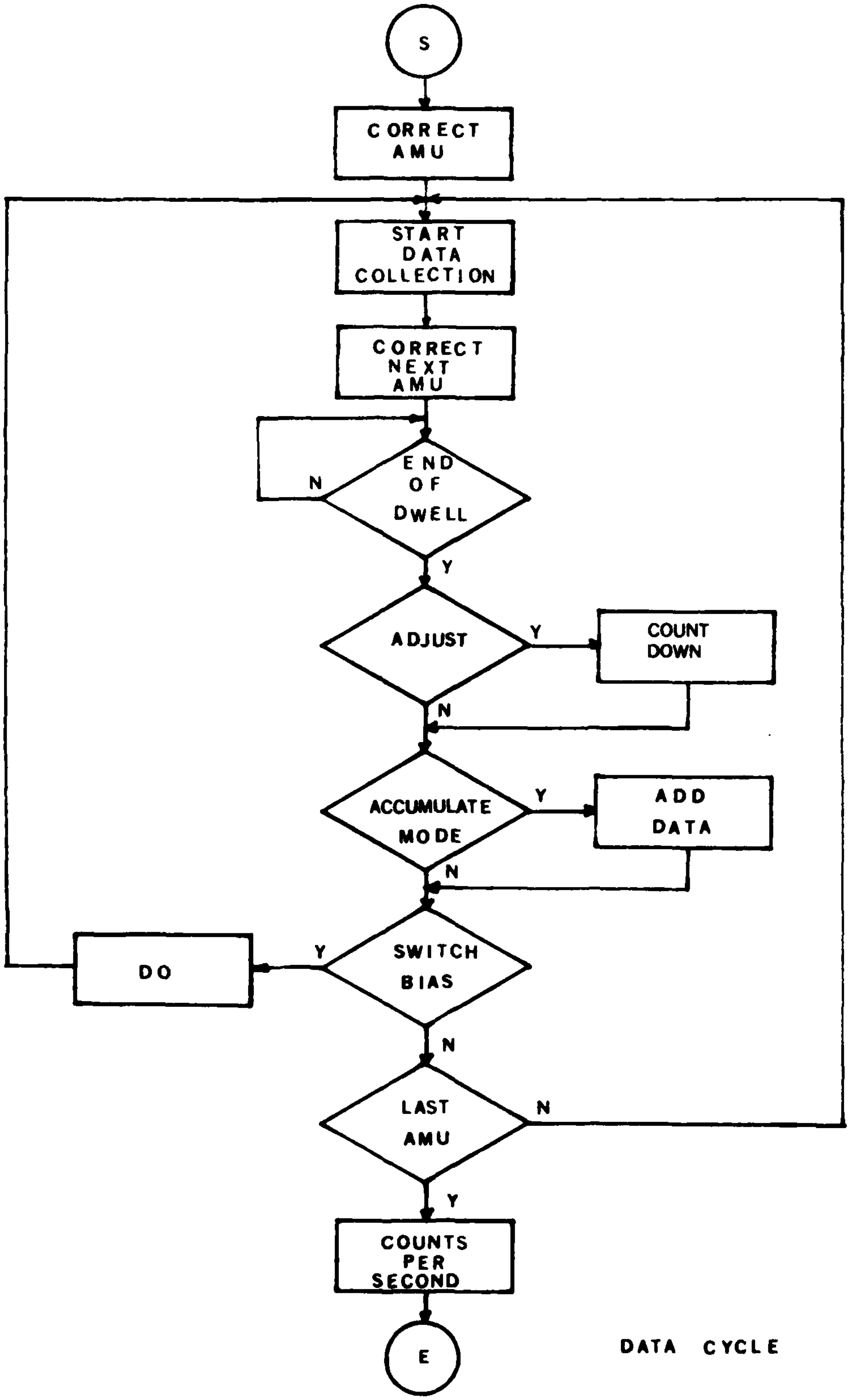


SYSTEM FLOW CHART

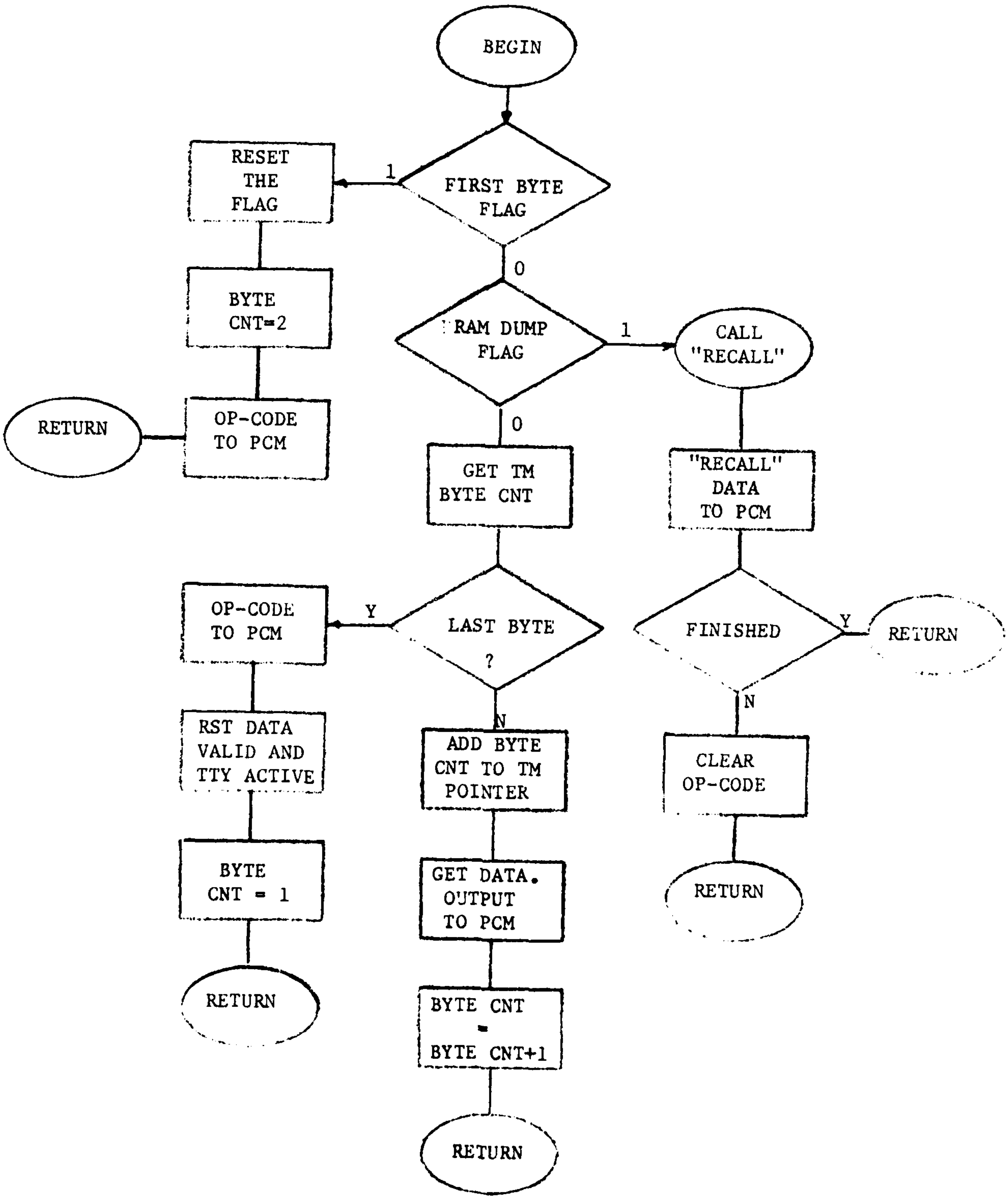


LIBRARY CYCLE

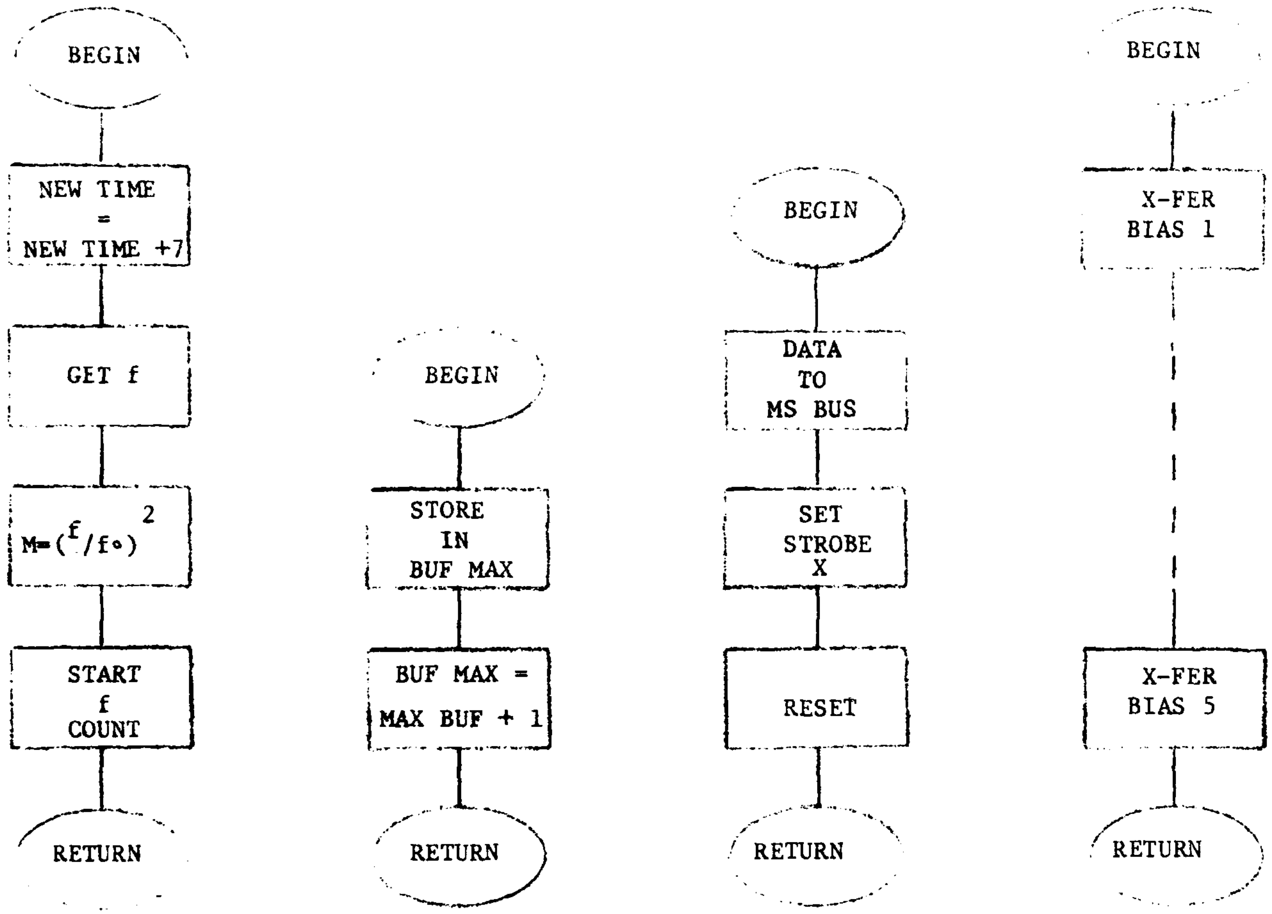

S
INQUIRE
REC. END TEXT
Y
Y
REC. START TEXT
N
Y
REQUEST REPEAT
RECEIVE COMMAND
N
2 OUT OF 3
N
OUT OF TIME
Y
Y
ECHO
RUN
Y
NEW INSTR.
N
RUN
Y
LIBR. INSTR.
N
DUMP RAM
Y
DUMP INSTR.
N
Y
WAIT
N
E
TTY LINK FLOW


S
CORRECT AMU
START DATA COLLECTION
CORRECT NEXT AMU
END OF DWELL
N
Y
ADJUST
Y
COUNT DOWN
N
ACCUMULATE MODE
Y
ADD DATA
N
SWITCH BIAS
Y
DO
N
LAST AMU
N
Y
COUNTS PER SECOND
E

DATA CYCLE

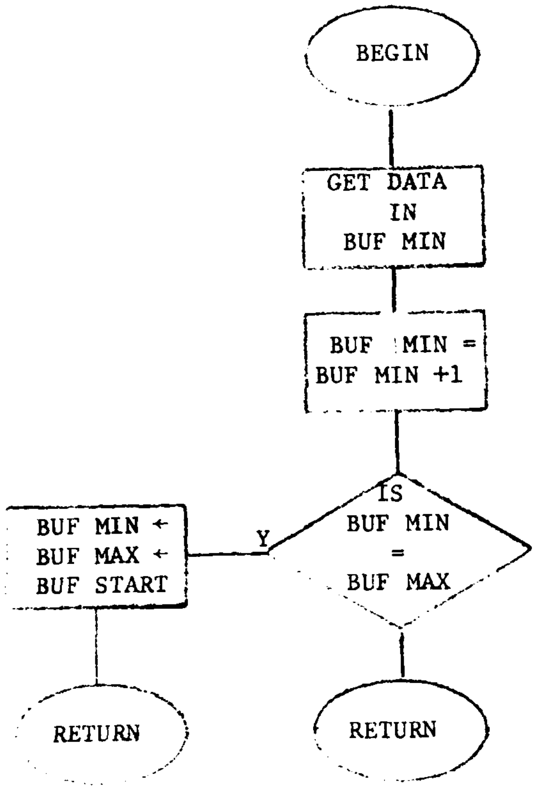


"INT 7.5" SUBROUTINE FLOW DIAGRAM

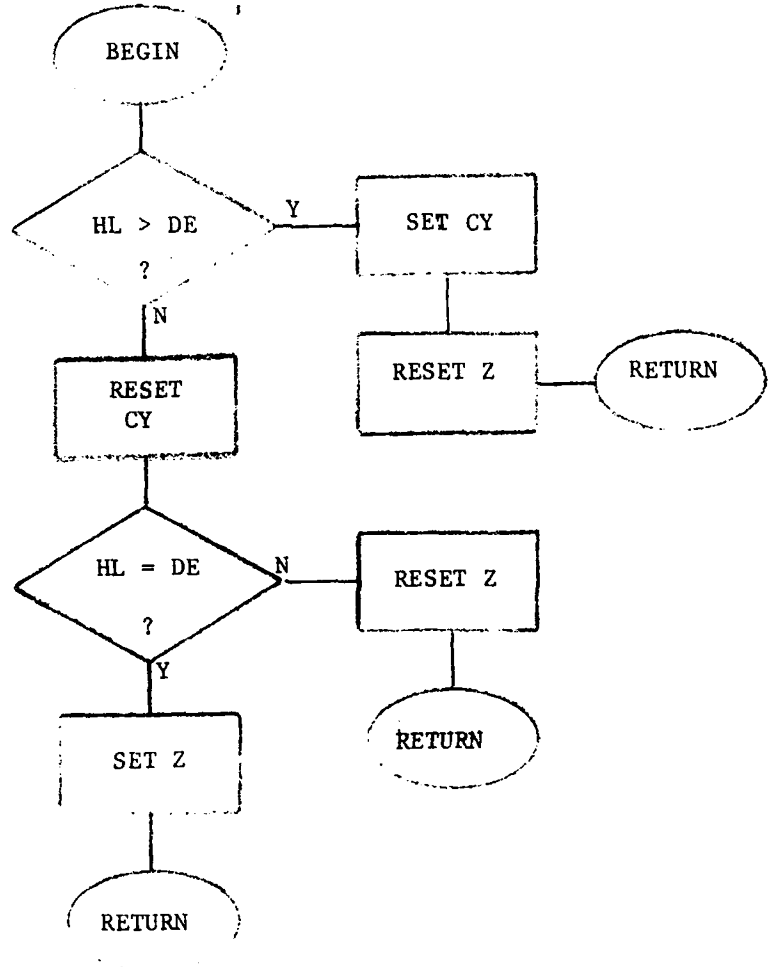


"FREQC" "STORE" "TRANSFR" "BIASOT"

FLOW DIAGRAMS

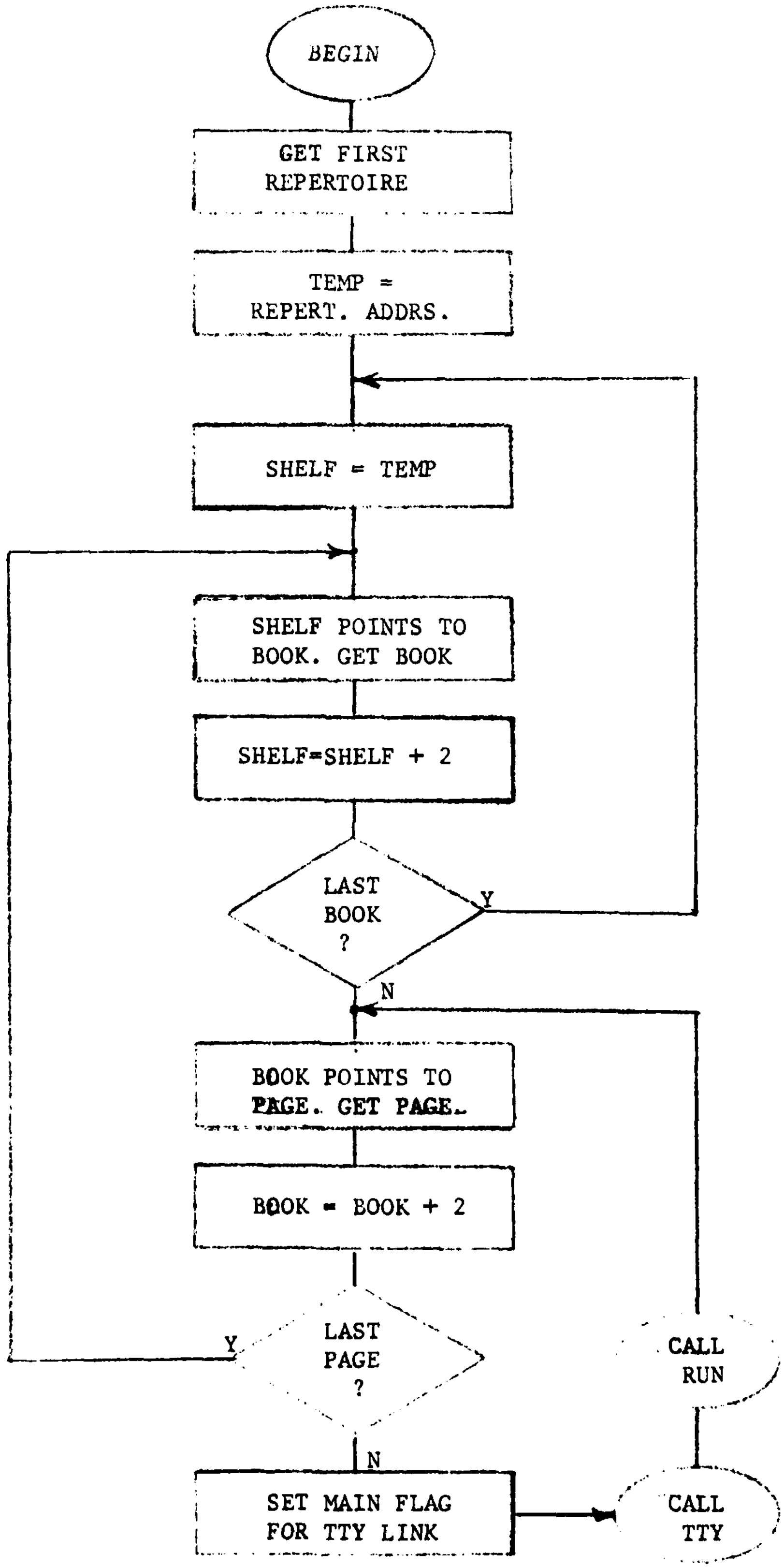


"RECALL"



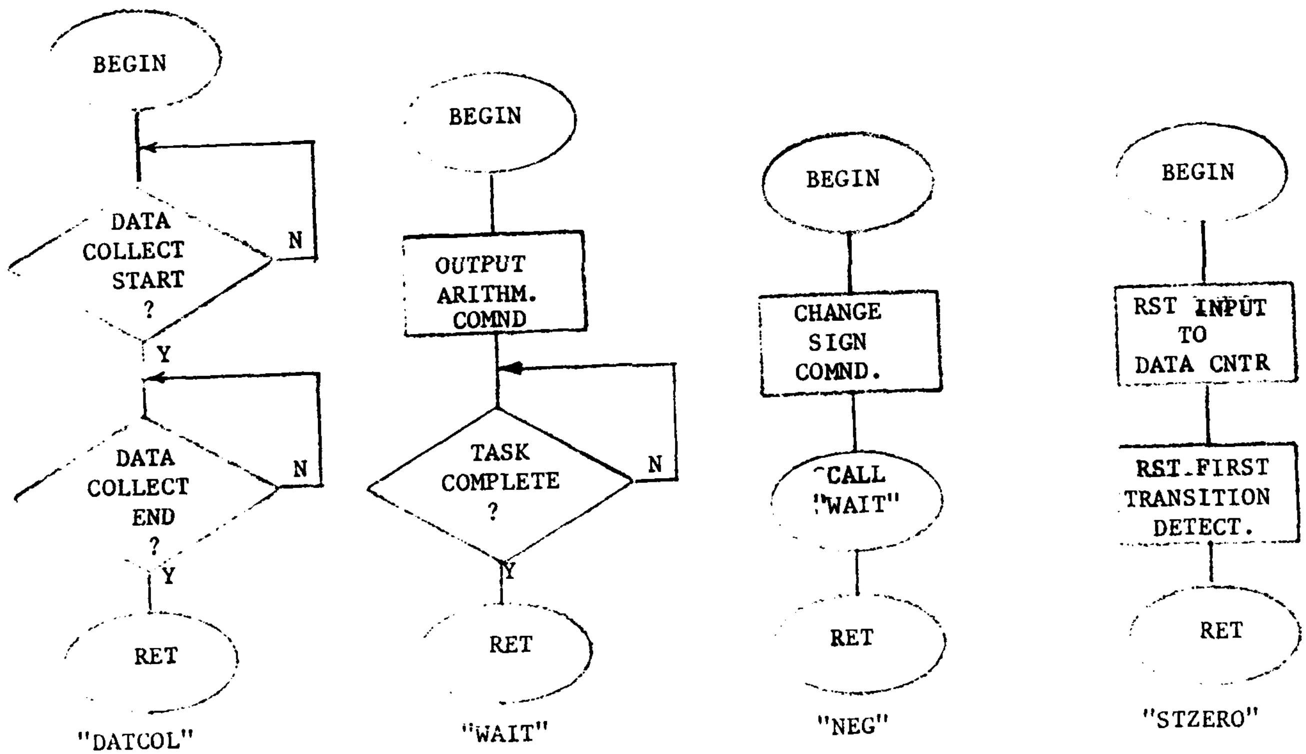
"CMPDH"

FLOW DIAGRAMS

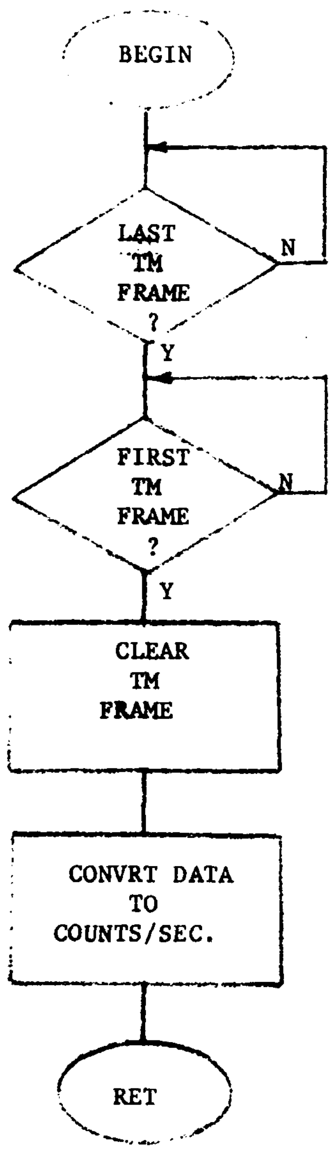


"MAIN"

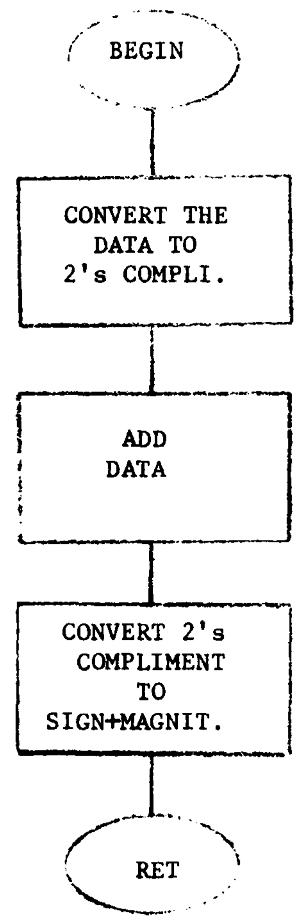
FLOW DIAGRAM



FLOW DIAGRAMS

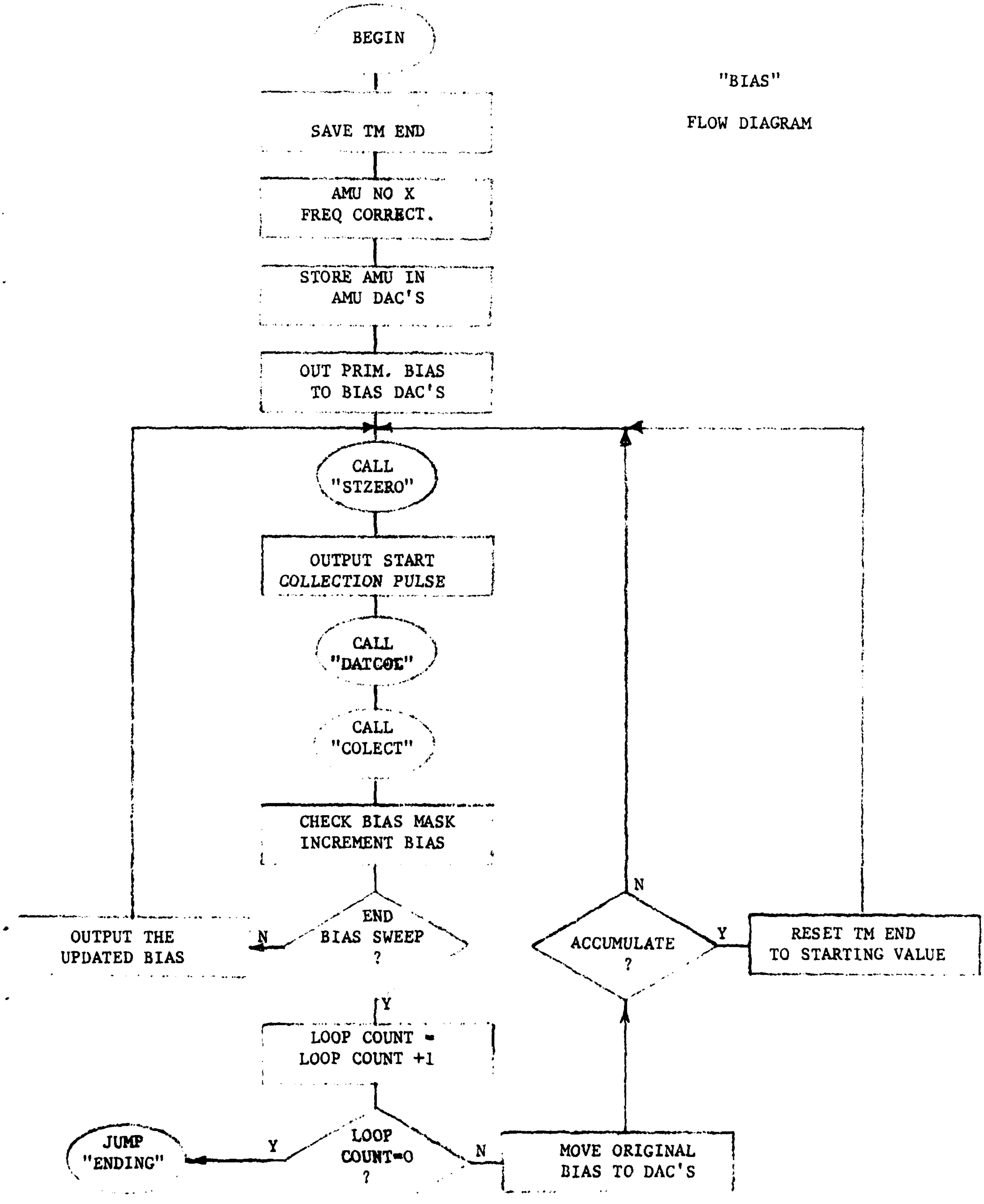


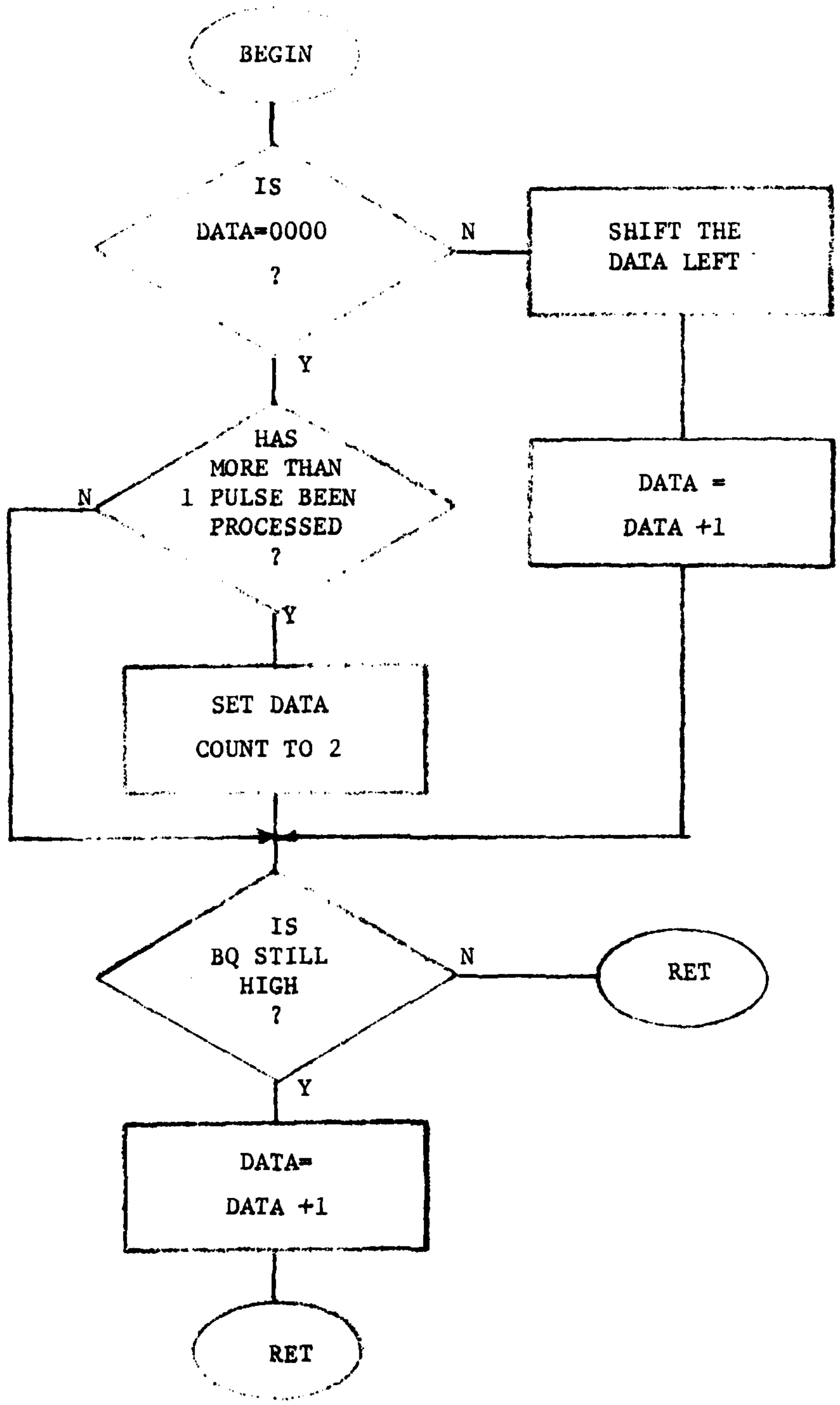
"ENDING"



"ADDDAT"

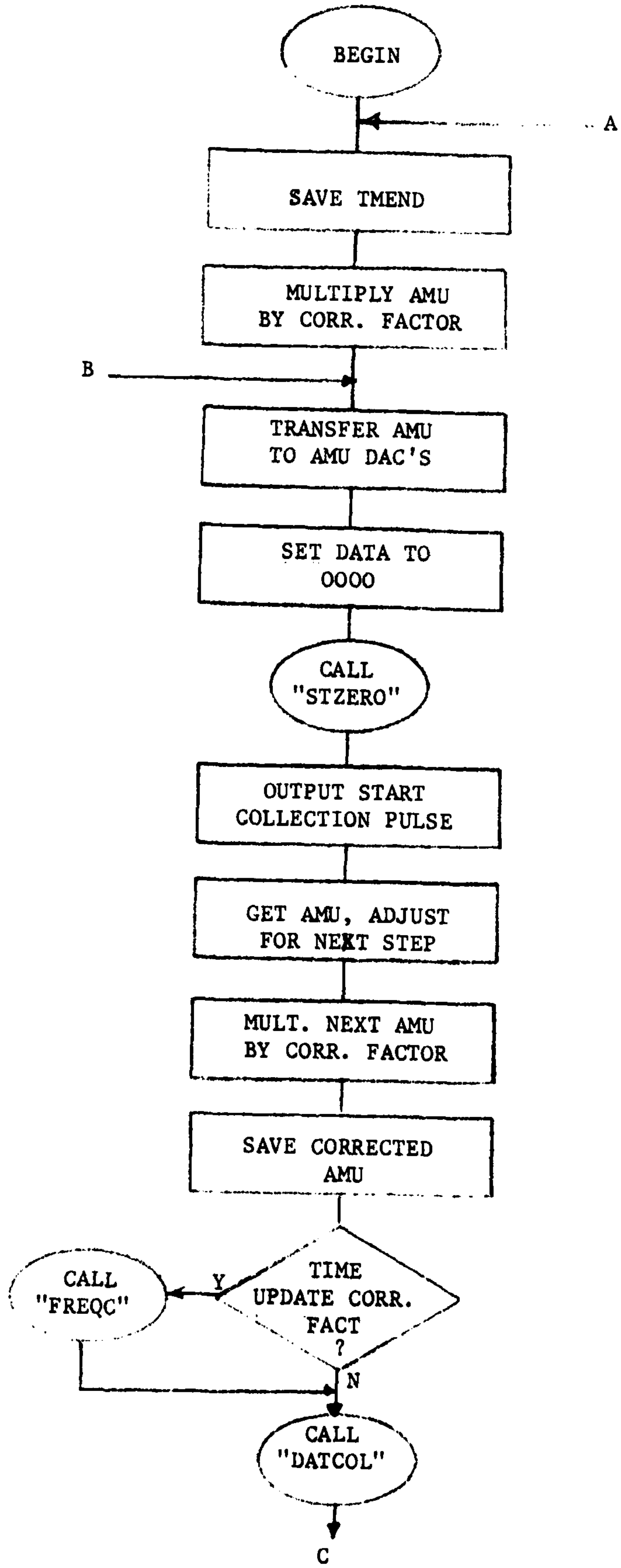
FLOW DIAGRAMS


BEGIN
"BIAS"
FLOW DIAGRAM
SAVE TM END
AMU NO X
FREQ CORRECT.
STORE AMU IN
AMU DAC'S
OUT PRIM. BIAS
TO BIAS DAC'S
CALL
"STZERO"
OUTPUT START
COLLECTION PULSE
CALL
"DATCOL"
CALL
"COLECT"
CHECK BIAS MASK
INCREMENT BIAS
END
BIAS SWEEP
?
N
OUTPUT THE
UPDATED BIAS
Y
LOOP COUNT =
LOOP COUNT +1
LOOP
COUNT=0
?
Y
JUMP
"ENDING"
N
MOVE ORIGINAL
BIAS TO DAC'S
ACCUMULATE
?
N
Y
RESET TM END
TO STARTING VALUE



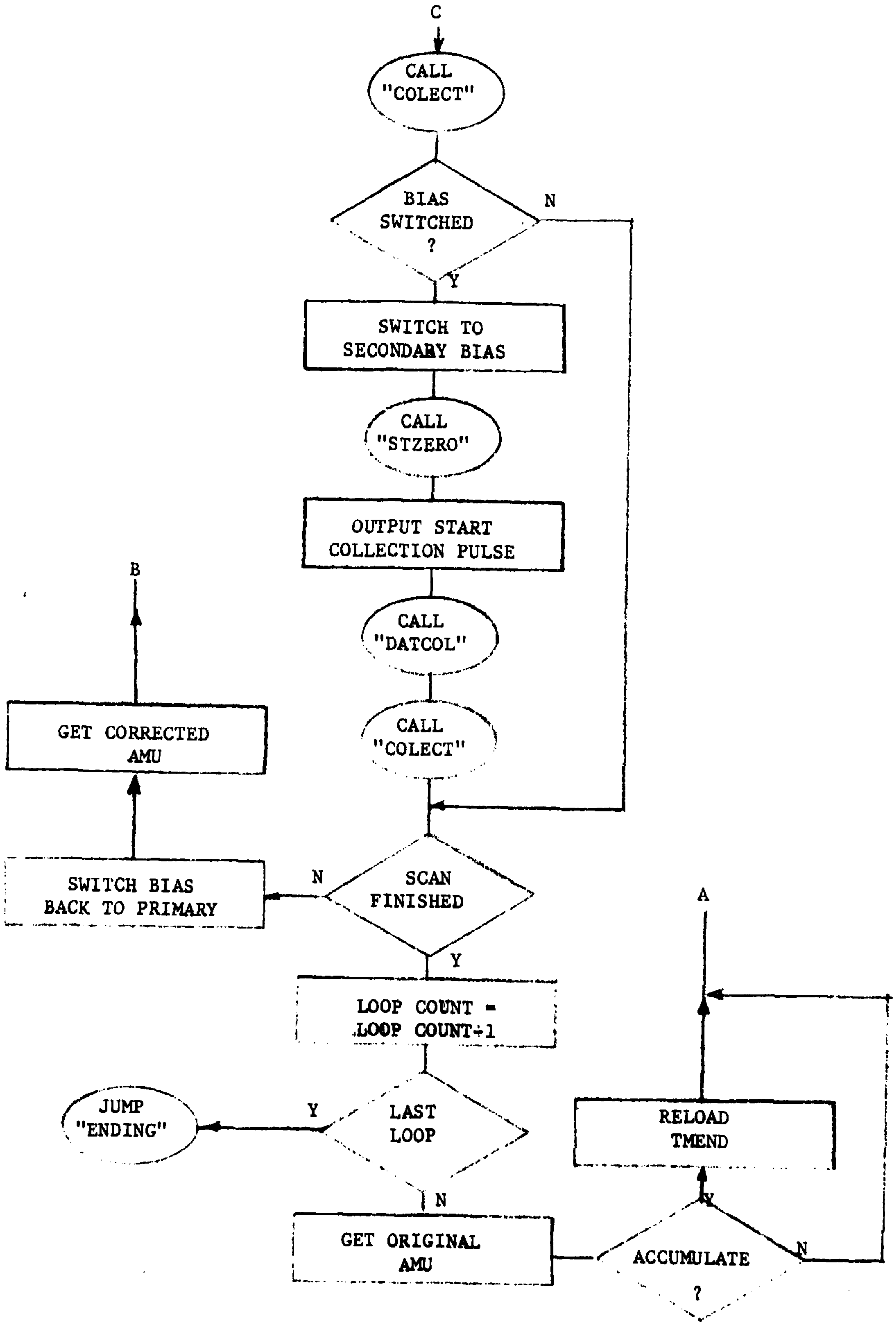
"CORDAT"

FLOW DIAGRAM

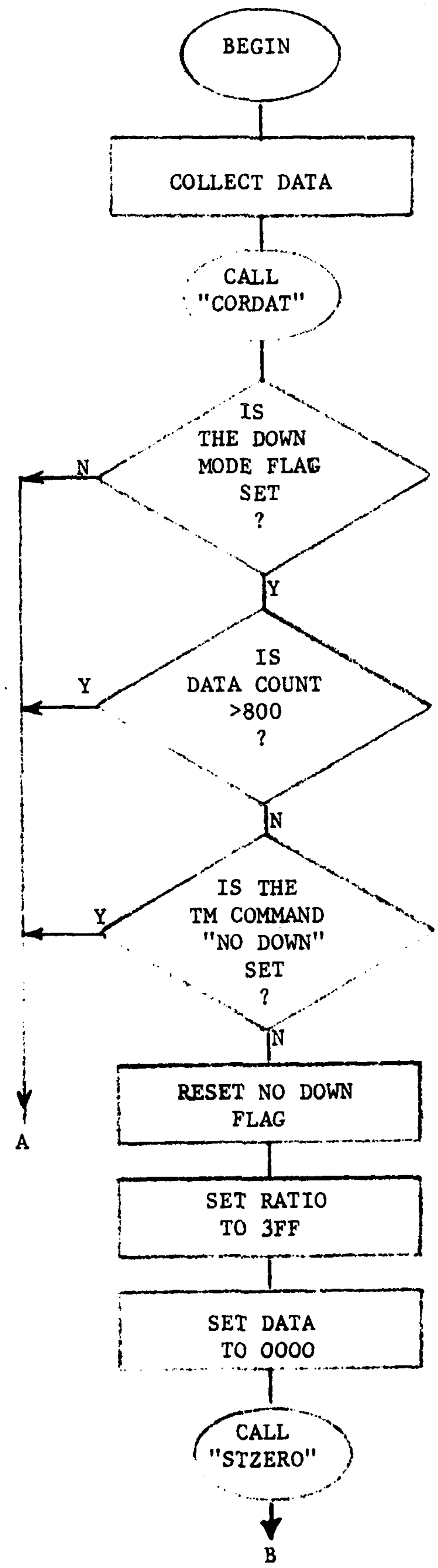


"AMU"

FLOW DIAGRAM

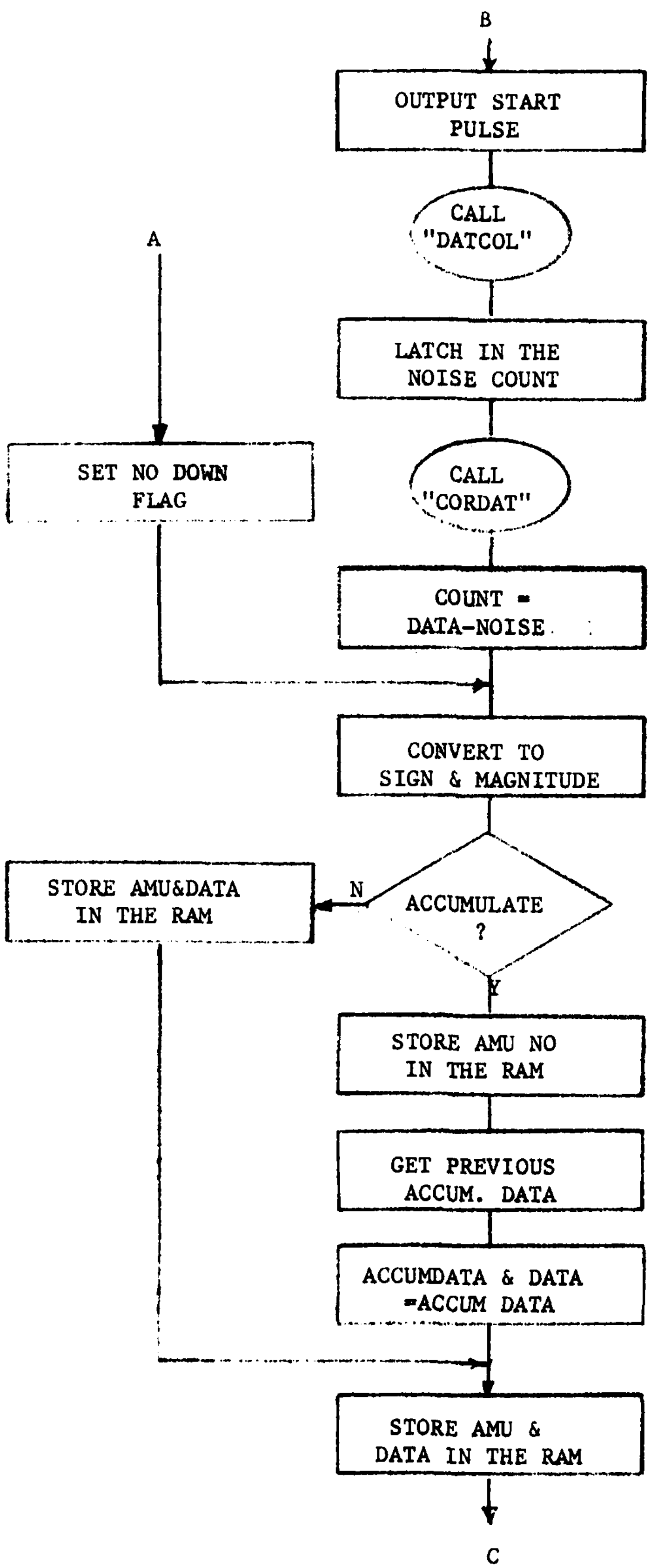


"AMU" FLOW DIAGRAM (CONT.)

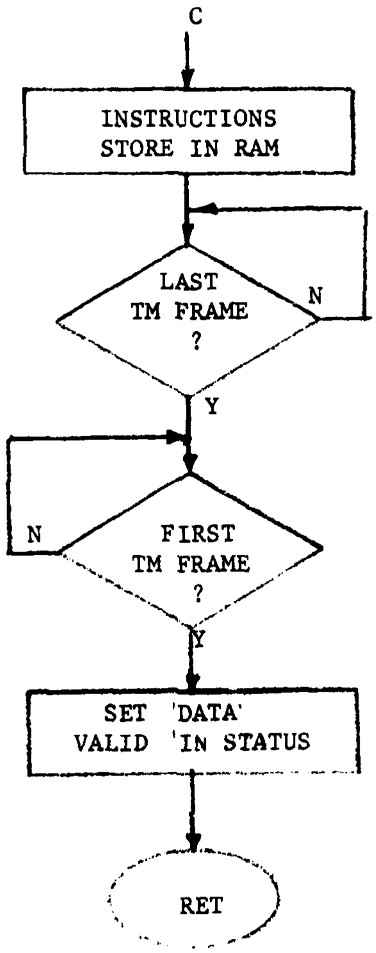


"COLLECT"

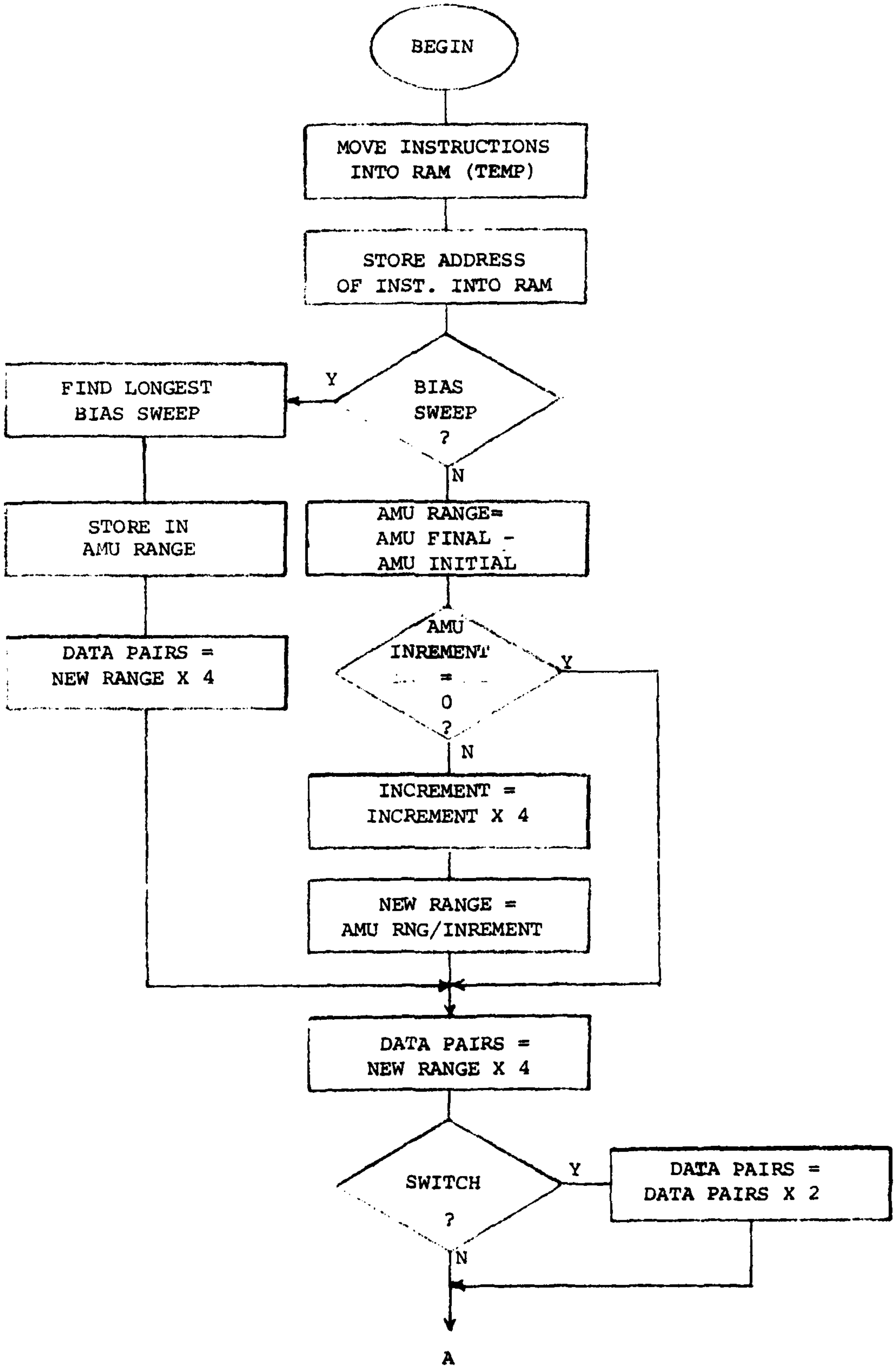
FLOW DIAGRAM



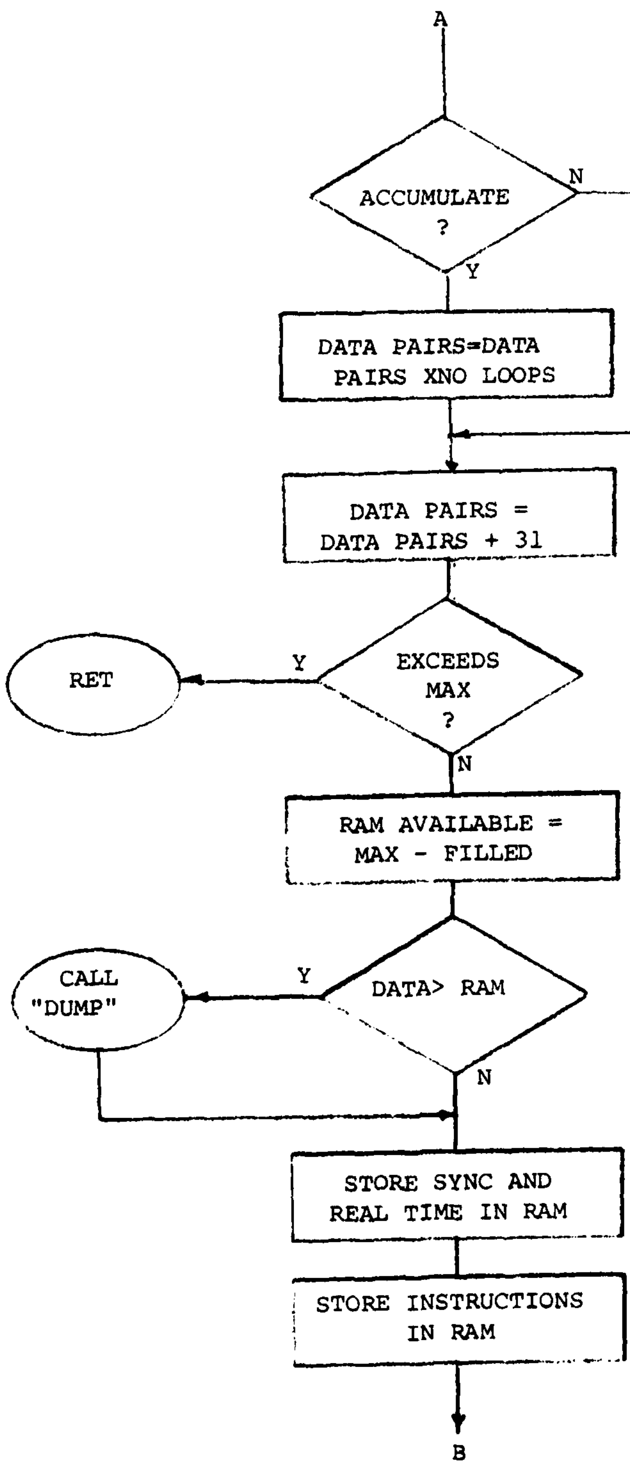
"COLLECT"
FLOW DIAGRAM
(CONT.)



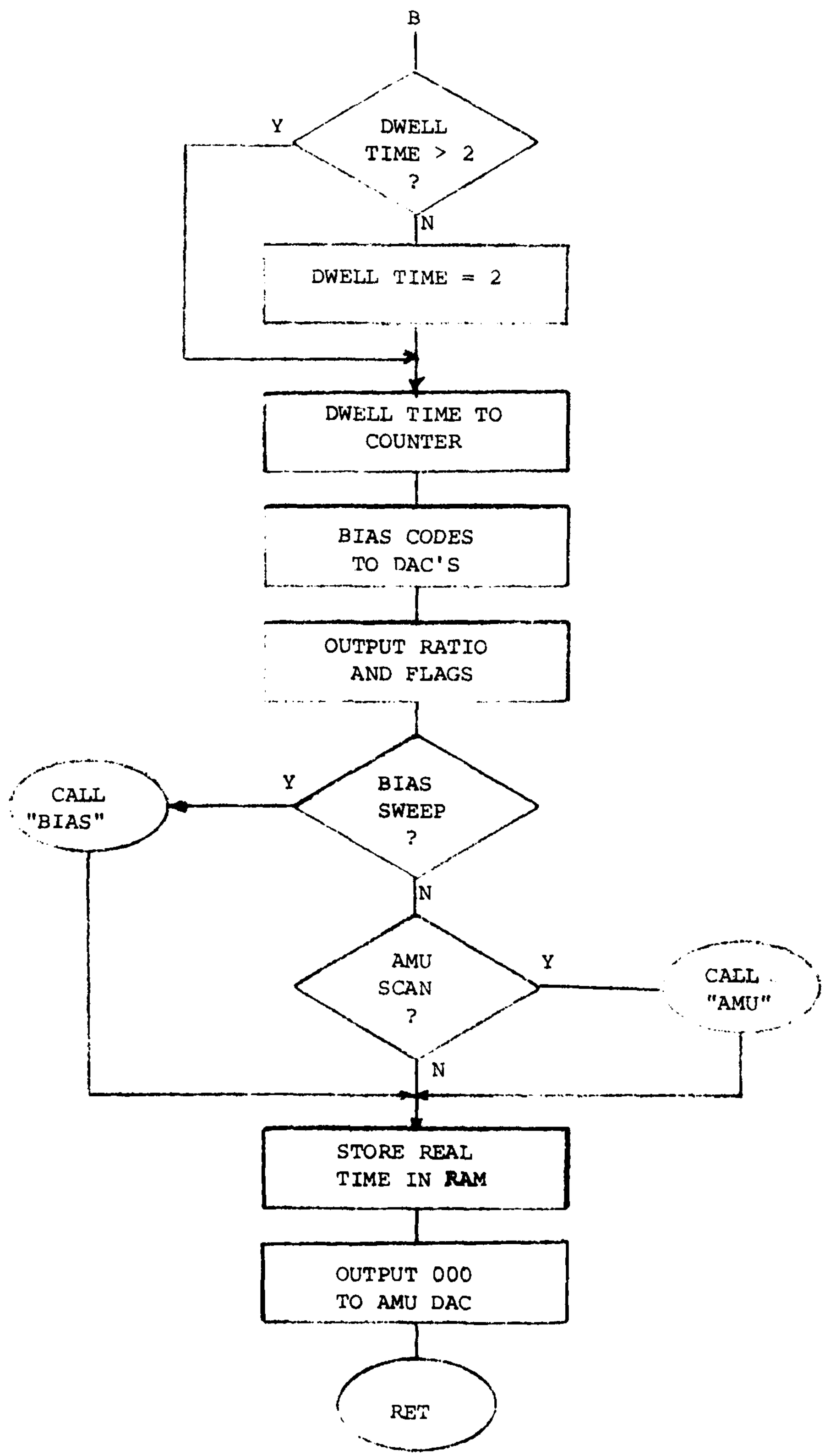
"COLLECT" FLOW DIAGRAM (CONT.)



"RUN" FLOW DIAGRAM



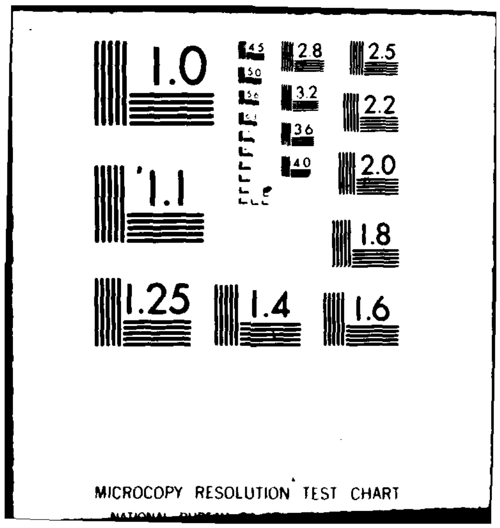
"RUN" FLOW DIAGRAM (CONT.)

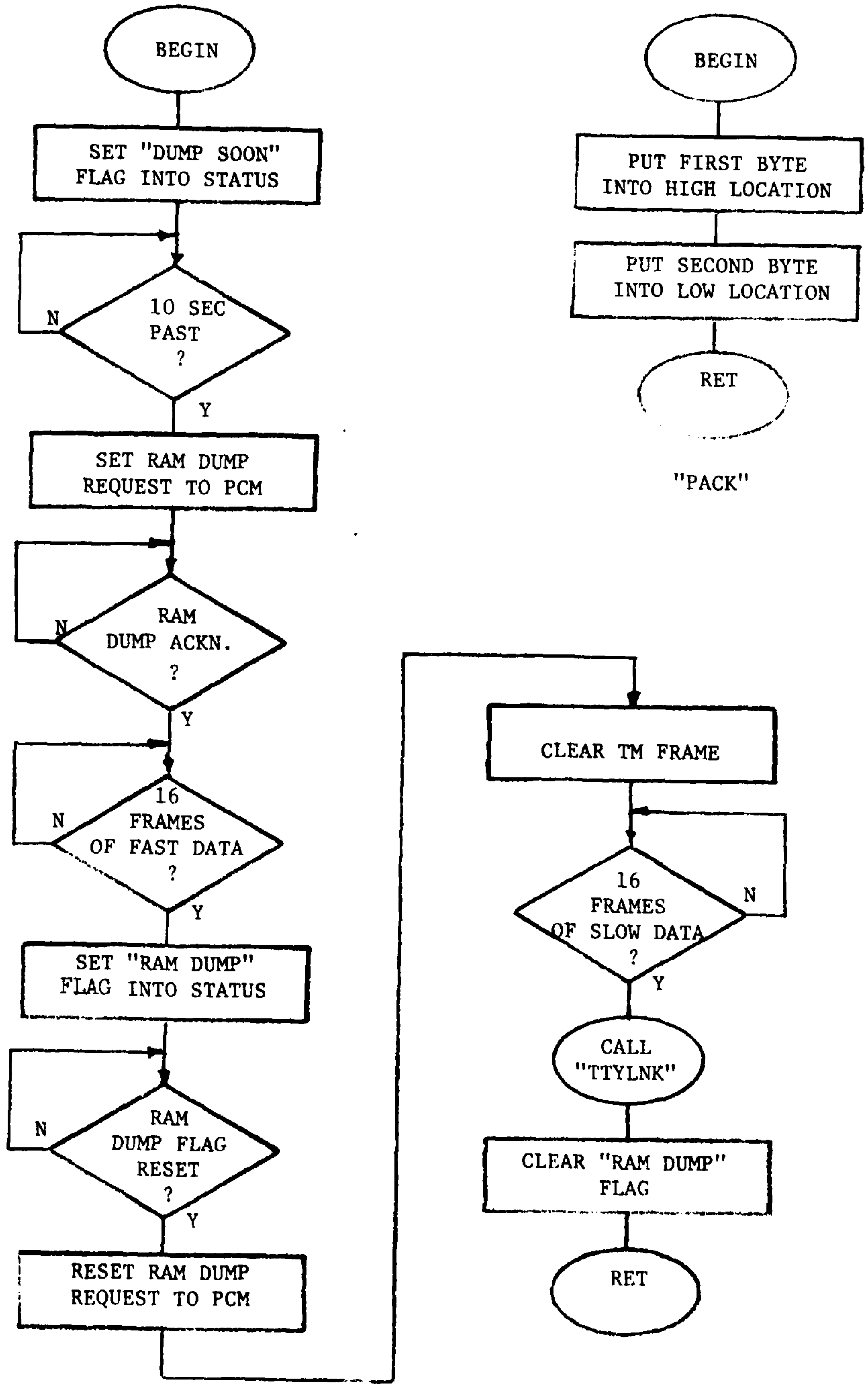


"RUN" FLOW DIAGRAM (CONT.)

AD-A115 399 NORTHEASTERN UNIV BOSTON MASS ELECTRONICS RESEARCH LAB F/G 7/4
CONTROL ELECTRONICS FOR AIR-BORNE QUADRUPOLE ION MASS SPECTROME--ETC(U)
OCT 81 J S ROCHEFORT, R SUKYS F19628-78-C-0218
UNCLASSIFIED AFGL-TR-82-0056 NL

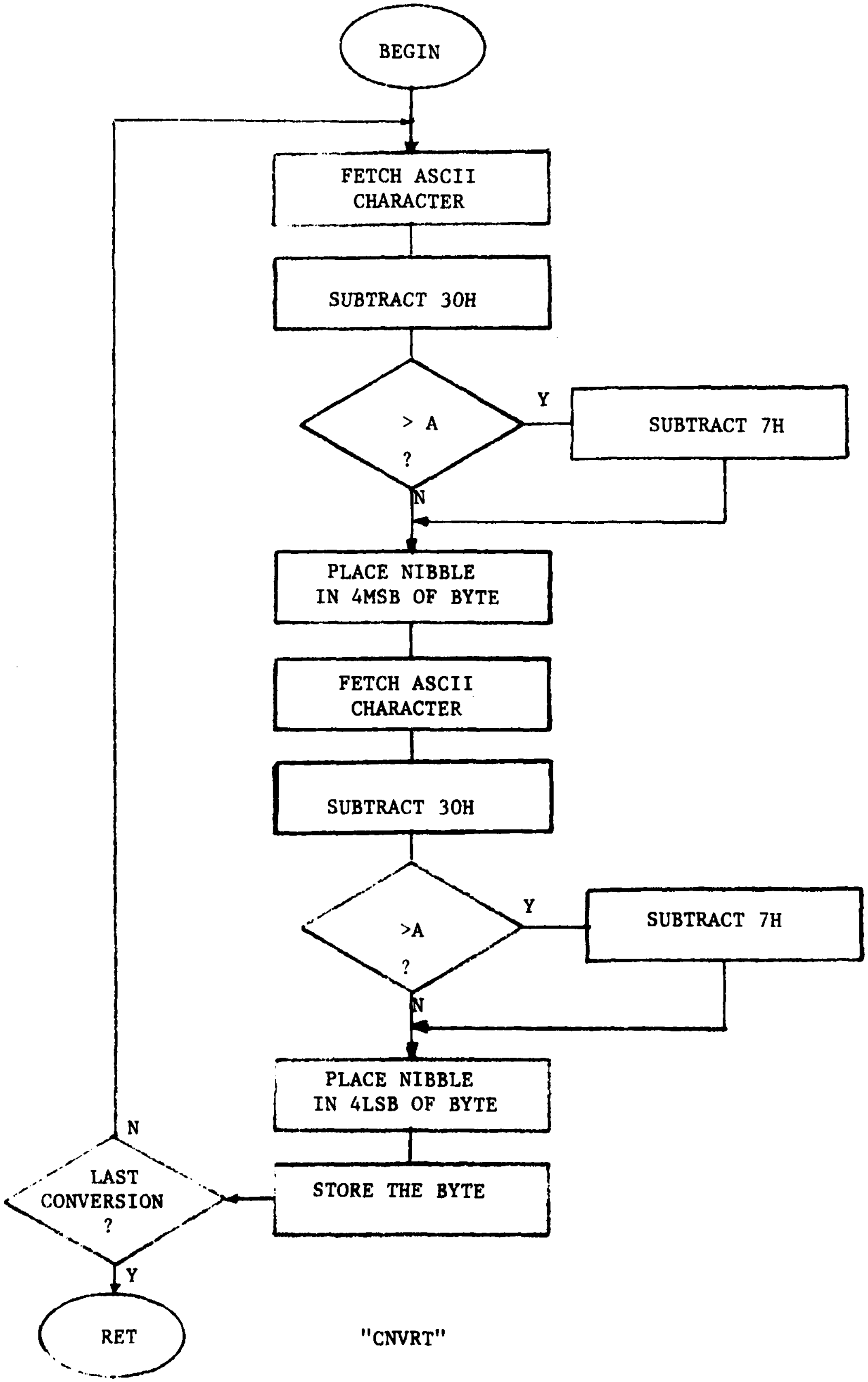
2 of 4


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART

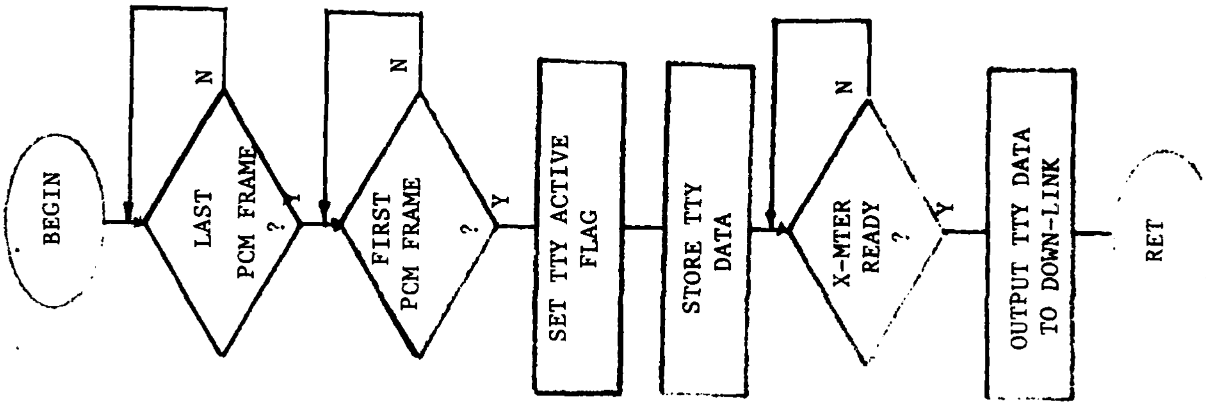

BEGIN
SET "DUMP SOON" FLAG INTO STATUS
10 SEC PAST ?
N
Y
SET RAM DUMP REQUEST TO PCM
RAM DUMP ACKN. ?
N
Y
16 FRAMES OF FAST DATA ?
N
Y
SET "RAM DUMP" FLAG INTO STATUS
RAM DUMP FLAG RESET ?
N
Y
RESET RAM DUMP REQUEST TO PCM
CLEAR TM FRAME
16 FRAMES OF SLOW DATA ?
N
Y
CALL "TTYLNK"
CLEAR "RAM DUMP" FLAG
RET
BEGIN
PUT FIRST BYTE INTO HIGH LOCATION
PUT SECOND BYTE INTO LOW LOCATION
RET

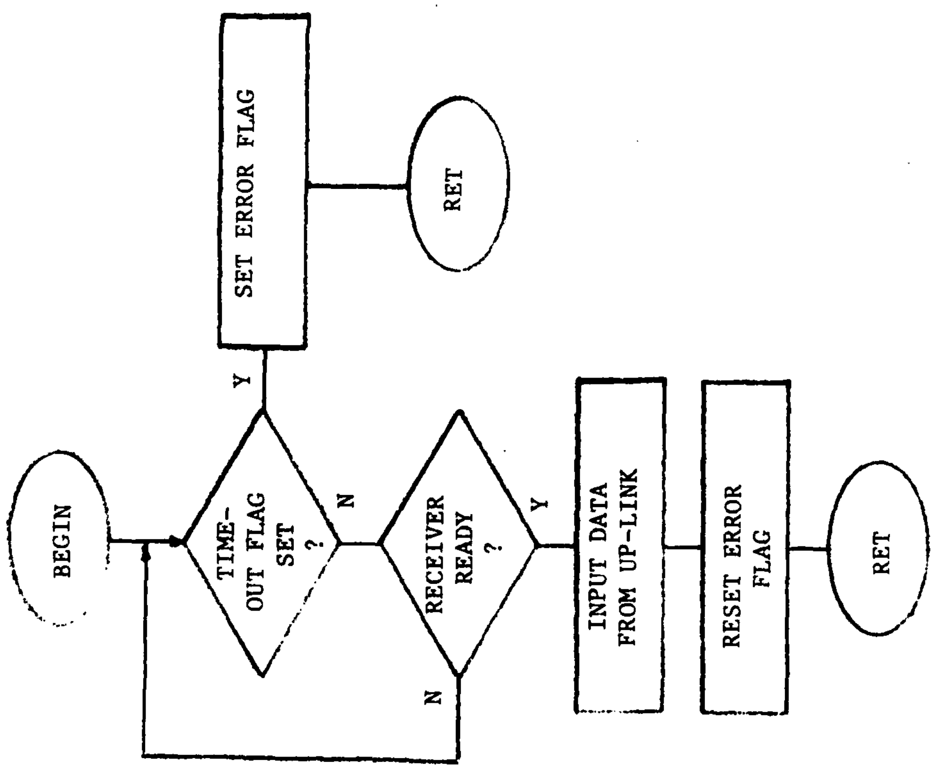
"PACK"

"DUMP


BEGIN
FETCH ASCII CHARACTER
SUBTRACT 30H
> A ?
Y
SUBTRACT 7H
N
PLACE NIBBLE IN 4MSB OF BYTE
FETCH ASCII CHARACTER
SUBTRACT 30H
>A ?
Y
SUBTRACT 7H
N
PLACE NIBBLE IN 4LSB OF BYTE
STORE THE BYTE
LAST CONVERSION ?
N
Y
RET

"CNVRT"


BEGIN
LAST PCM FRAME ?
N
Y
FIRST PCM FRAME ?
N
Y
SET TTY ACTIVE FLAG
STORE TTY DATA
X-MTER READY ?
N
Y
OUTPUT TTY DATA TO DOWN-LINK
RET


BEGIN
TIME-OUT FLAG SET ?
Y
N
SET ERROR FLAG
RET
RECEIVER READY ?
Y
N
INPUT DATA FROM UP-LINK
RESET ERROR FLAG
RET

"RECEV"

INT 7.5

TYPE: INTERUPT.

ENTER: NO CONDITIONS.

RETURN: REGISTER CONTENTS NOT AFFECTED.

COMMENT: INTERUPT VECTOR INT7.5 MUST BE SET AND INTERUPTS MUST BE ENABLED.

MEMORY: 802F: STATUS WORD.
802E: TM BYTE COUNT.

I/O PORTS: 5B ; "FIRST FRAME":FLAG STATUS.
44 : TM PORT.
59 ; "FIRST FRAME" FLAG RESET.

MAIN

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: CONTENTS OF ALL REGISTERS AFFECTED.

MEMORY: 8025-6 ; TEMP. STORAGE OF PROGRAM POINTER.
8027-8 ; PROGRAM POINTER.
8029-A ; INSTRUCTION SET POINTER.
8065 ; "ALL COMMAND FUNCTION" FLAG.

FREQC

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: ACC & FLAGS AFFECTED.

MEMORY: 8020 ; THE TIME TO CALL FREQC AGAIN.
805A-805D ; FREQUENCY CORRECTION FACTOR.

I/O PORTS: 64 : FREQ. COUNTER.
54 : ARITHMETIC UNIT.
59 : START PULSE FOR FREQ. COUNTER.

TRNSFR

TYPE: SUBROUTINE.

ENTER: ACC CONTAINS DATA TO BE OUTPUTED TO MS BUS.
B CONTAINS ADDRESS WHERE MS BUS DATA IS TO BE SENT.

RETURN: ACC-CLEARED, CY=0, AC=0, Z=1, S=0, P=1.

I/O PORTS: 40 ; MS BUS.
50 ; MS ADDRESS.

BIASOT

TYPE: SUBROUTINE.

ENTER: E CONTAINS DATA FOR BIAS1.
D CONTAINS DATA FRO BIAS2.
L CONTAINS DATA FOR BIAS3.
H CONTAINS DATA FOR BIAS4.
C CONTAINS DATA FOR BIAS5.

RETURN: ACC-CLEARED
B -CONTAINS ADDRESS OF BIAS5.
CY=AC = S = 0.
Z = P = 1.

RECALL

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: ACC CONTAINS DATA FROM DUMP BUFFER.
HL CONTAINS NEXT DUMP BUFFER LOCATION.
FLAGS AFFECTED.

MEMORY: 8023-4 ;CONTAINS BEGINNING OF DUMP BUFFER ADDRESS.
8021-2 ;CONTAINS END OF DUMP BUFFER ADDRESS.

STORE

TYPE: SUBROUTINE.

ENTER: ACC CONTAINS DATA TO BE STORED IN DUMP BUFFER.

RETURN: REGISTER CONTENTS NOT AFFECTED.

MEMORY: 8021-2 :LOCATION OF DUMP BUFFER POINTER.
8100-MAX MEMORY: DUMP BUFFER.

WAIT

TYPE: SUBROUTINE.

ENTER: ACC CONTAINS A COMMAND TO ARITHMETIC UNIT.

RETURN: ACC AFFECTED.
CY=0

I/O PORTS: 55 ;ARITHMETIC UNIT.

COLECT

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: ACC CONTAINS LOOP COUNTER COUNT.
HL CONTAINS LAST TM FRAME ADDRESS.

CONTENTS OF OTHER REGISTERS ALSO AFFECTED.

MEMORY:	8020	;"DOWN COUNT" FLAG.
	8009	;INSTRUCTION SET MODE.
	801C-D	;PRESENT AMU.
	8021-2	;END OF DUMP BUFFER ADDRESS.
	802F-3B	;DATA FOR PCM.
	802E	;TM BYTE COUNTER.
	8016-7	;ID
	8005-6	;DWELL TIME.
	8007	;RATIO (LOW BYTE).

I/O PORTS:	63	;COMMANDS FOR DATA COUNTER.
	61	;DATA COUNTER.
	5A	;"NO DOWN" FLAG.
	50	;"START COLLECTION" PULSE.

DATCOL

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: ACC AFFECTED, Z=0, P=1, CY=0, AC=0, S=0.

I/O PORTS: 5B ;DATA COUNTER ENABLE MONITOR

STZERO

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: PSW AFFECTED.

```
I/O PORTS:  5B      ;MS DATA INPUT MONITOR.
            59      ;MS DATA INPUT COMPLIMENTER.
```

CORDAT

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: HL CONTAINS DATA FROM MS DATA COUNTER.
PSW AFFECTED.

```
I/O PORTS:  5B      ;MS INPUT MONITOR.
```

TTYLNK

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: PSW, DE, and BC AFFECTED.

```
MEMORY:  5800-5808  ;TTY BUFFERS.
         8065       ;"LIMITED COMMAND" FLAG.
         8025-6     ;REPERTOIRE POINTER.
         8027-8     ;TEMP. REPERTOIRE POINTER.
         8029-A     ;INSTRUCTION SET POINTER.
         8067       ;"WAIT" FLAG.
         802E       ;TM BYTE COUNTER.
         8000-17    ;INSTRUCTION SET PARAMETERS.

I/O PORTS:  69      ;USART COMMAND.
            65      ;"TIME-OUT" TIMER.
            68      ;USART.
```

PACK

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: HL CONTAINS PACKED DATA.
ACC AFFECTED.

```
MEMORY:  5802       ;HI BYTE ADDRESS.
         5803       ;LO BYTE ADDRESS.
```

NEG

TYPE: SUBROUTINE.

ENTER: CY=1.
ACC SHIFTED LEFT BY ONE WHEN COMPARED TO H.
HL CONTAINS DATA.

RETURN: HL CONTAINS POSITIVE DATA.
ACC AFFECTED.
CY=0.

BIAS

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

MEMORY:	8021-2	;END OF RAM DUMP BUFFER.
	8057-8	;TEMP STORAGE OF END OF RAM DUMP BUFFER.
	8000-1	;STARTING AMU NUMBER.
	801C-D	;TEMP STORAGE OF STARTING AMU NUMBER.
	805A-D	;FREQ. CORRECTION FACTOR.
	800A-E	;PRIMARY BIASES.
	803F-43	;TEMP. STORAGE OF PRIMARY BIASES.
	8014	;BIAS MASK.
	800F-13	;SECONDARY BIASES.
	8004	;CONTENTS OF LOOP COUNTER.
	8009	;MODE OF INSTRUCTION SET.

I/O PORTS:	54	;ARITHMETIC UNIT.
	50	;START OF COLLECTION PULSE.

ADDDAT

TYPE: SUBROUTINE.

ENTER: HL CONTAINS DATA, SIGNED MAGNITUDE.
DE CONTAINS DATA, SIGNED MAGNITUDE.

RETURN: HL = HL+DE, SIGNED MAGNITUDE.

AMU

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

```
MEMORY:  8021-2     ;END OF RAM DUMP BUFFER.
         8057-8     ;TEMP. STORAGE OF END OF RAM DUMP BUFFER.
         8000-1     ;STARTING AMU NUMBER.
         802B-C     ;TEMP. STORAGE OF STARTING AMU NUMBER.
         805A-D     ;FREQ. CORRECTION FACTOR.
         801C-D     ;TEMP. STORAGE OF PRESENT AMU NUMBER.
         8015       ;STEP VALUE.
         803D-E     ;TEMP. STORAGE OF NEXT AMU NUMBER, CORRECTED.
         8020       ;THE TIME TO CALL "FREQC" AGAIN.
         8009       ;MODE OF INSTRUCTION SET.
         800A-13    ;PRIMARY AND SECONDARY BIASES.
         8002-3     ;FINAL AMU NUMBER.
         8004       ;NUMBER OF TIMES TO LOOP.

I/O PORTS:  54      ;ARITHMETIC UNIT.
            63      ;REAL TIME COUNTER COMMAND.
            62      ;REAT TIME COUNTER.
            50      ;START OF COLLECTION PULSE.
```

ENDING

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

```
MEMORY:  802E       ;TM BYTE COUNT.
         802F-3B    ;PCM FRAME DATA.
         8005-6     ;DWELL TIME.
         8057-8     ;TEMP. STORAGE OF END OF RAM DUMP BUFFER.
         8009       ;MODE  OF INSTRUCTION SET.
         8059       ;TEMP. LOOP COUNT COUNTER.
         8021-2     ;END OF RAM DUMP BUFFER.

I/O PORTS:  54      ;ARITHMETIC UNIT.
```

CMPDH

TYPE: SUBROUTINE.

ENTER: DE AND HL VALUES TO BE COMPARED.

RETURN: ACC EFFECTED.
CY=1, Z=0 HL > DE.
CY=0, Z=0 HL < DE.
CY=0, Z=1 HL = DE.
REST OF FLAGS AFFECTED.

DUMP

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: REGISTER CONTENTS AFFECTED.

MEMORY: 802F ;PCM STATUS WORD.
802E ;TM BYTE COUNT.
8030-3C ;MS FRAME DATA.
8065 ;"LIMITED COMMAND FUNCTION" FLAG.
8061-2 ;CONTAINS MAX MEMORY ADDRESS.
8100-MAX MEMORY; DUMP BUFFER.

I/O PORTS 65 ;"TIME-OUT" TIMER.
5B ;"TIME-OUT" FLAG .
59 ;"RAM DUMP REQUEST" FLAG.

RUN

TYPE: SUBROUTINE.

ENTER: HL CONTAINS BEGINNING OF THE INSTRUCTION SET TO BE RUN.

RETURN: ALL REGISTERS AFFECTED.

MEMORY: (HL) -[(HL) +15] ;INSTRUCTION SET TO BE RUN.
8000-17 ;INSTRUCTION SET BEING RUN.
8061-2 ;CONTAINS ADDRESS OF MAX DUMP BUFFER.
8059 ;TEMPORARY LOOP COUNTER STORAGE.
6000 ;DWELL TIME TIMER.

I/O PORTS: 63 ;COMMAND FOR REAL TIME COUNTER.
62 ;REAL TIME COUNTER.

CNVRT

TYPE: SUBROUTINE.

ENTER: HL CONTAINS STARTING ADDRESS OF CONVERSION.
DE CONTAINS END ADDRESS OF CONVERSION.

RETURN: DE, B, and PSW AFFECTED.

TRNSMT

TYPE: SUBROUTINE.

ENTER: ACC CONTAINS THE BYTE TO BE TRANSMITTED.

RETURN: FLAGS AFFECTED.

```
MEMORY:  802E       ;TM BYTE COUNTER
         802F       ;STATUS BYTE.
         8037       ;TTY DOWN LINK LOCATION.

I/O PORTS:   69     ;USART COMMAND.
             68     ;USART
```

RECEV

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: ACC CONTAINS RECEIVED CHARACTERS, IF CY=0
CY=1 IF TIME-OUT OCCURRED.
CY=0 IF ACC IS OK.
REST OF FLAGS ALSO AFFECTED.

```
I/O PORTS:  5B      ;"TIME OUT" FLAG.
            69      ;USART COMMAND.
            68      ;USART.
```

FLAGS

8065 "LIMITED COMMANDS ONLY" FLAG.

IF TTY LNK IS USED AND THIS FLAG IS SET THEN ONLY THE COMMAND "DUMP", "WAIT" AND "CONTINUE" ARE ENABLED.

8067 "WAIT" FLAG.

IF THIS FLAG IS SET, THEN THE LAST COMMAND WAS A "WAIT" COMMAND.

RAM

5800-5842	UP LINK BUFFER #1.
5843-5885	UP LINK BUFFER #2.
5886058C8	UP LINK BUFFER #3.
58C9-58FF	STACK.
8000	INITIAL AMU LOW BYTE.
8001	INITIAL AMU HIGH BYTE
8002	FINAL AMU LOW BYTE
8003	FINAL AMU HIGH BYTE.
8004	NUMBER OF TIMES IN LOOP.
8005	DWELL TIME LOW BYTE.
8006	DWELL TIME HIGH BYTE.
8007	RATION LOW BYTE.
8008	RATIO HIGH BYTE.
8009	MODE.
800A-8013	PRIMARY AND SECONDARY BIASES.
8014	BIAS SWEEP MASK.
8015	STEPPING VALUE.
8016-8017	PROGRAM ID.
8018-801B	SPARE.
801C-801D	PRESENT AMU.
801E	FREQUENCY UPDATE TIME.
8021-8022	TMEND.
8023-8024	TMBGN.
8025-8026	REPERTOIRE POINTER.
8027-8028	REPERTOIRE POINTER BUFFER.
8029-802A	PROGRAM POINTER.
802B-802C	ORIGINAL AMU BUFFER.
802D	"NO DOWN COUNT" FLAG.
802E	TM WORD COUNTER.
802F	STATUS OF INSTRUCTIN SET BEING RUN.
8030-8031	ID OF INSTRUCTION SET BEING RUN.
8032-8033	DATA OF AMU.
8034-8035	DWELL TIME OF INSTRUCTION SET.
8036	RATIO OF INSTRUCTION SET BEING RUN.
8037	TTY DOWN LINK DATA.
8038-8039	AMU BEING DETECTED.
803A	MODE OF INSTRUCTION SET BEING RUN.

803B	LOOP COUNTER OF INSTRUCTION SET.
803C	SPARE.
803D-803E	NEXT CORRECTED AMU.
803F-8043	ORIGINAL PRIMARY BIASES.
8044-8045	SPARES.
8046-8056	SPARES.
8057-8058	TMEND AT START OF A INSTRUCTION SET.
8059	ORIGINAL LOOP COUNTER OF INSTRUCTION SET BEING RUN.
805A-805D	FREQUENCY CORRECTION FACTOR (MSB OF MSW FIRST).
805E-8060	SPARE.
8061-8062	MAXIMUM MEMORY LOCATION.
8063-8064	SPARE.
8065	"LIMITED COMMAND ONLY" FLAG.
8066	SPARE.
8067	"WAIT""FLAG
8068-80FF	SPARE.
8100-MAX MEMORY	RAM DUMP DATA.

EPROM

0000-0FFF	SYSTEM PROGRAMS.
1000-1FFF	REPERTOIRES, PROGRAMS, INSTRUCTION SETS.

I/O PORTS

40	MS DATA BUS.
44	PCM DATA BUS.
50	STROBES AND START PULSE.
54	ARITHMETIC PROCESSOR.
55	COMMAND FOR ARITHMETIC PROCESSOR.
58	COMMAND FOR THE 5 PORTS IN 59, 5A, 5B, 5C and 5D.
59	FUNCTION FLAGS.
5A	SPARE
5B	MONITORS.
5C	LOW BYTE OF USART CLOCK.
5D	HIGH BYTE OF USART CLOCK.
60	DWELL TIMER.
61	DATA COUNTER.
62	FLIGHT TIMER.
63	COMMAND FOR THE 3 PORTS IN 60, 61 and 62.
64	FREQUENCY CORRECTION COUNTER.
65	SPARE.
66	SPARE.
67	COMMAND FOR THE 3 PORTS IN 64, 65 and 66.

```
        $TITLE   "FLIGHT PROGRAM FOR BBIMS WRITTEN BY JIM MANLEY"
        GLOBAL   TMBYT2,FRM1,FRM16
TMBYT1  EQU      0EH       ;SET NUMBER OF BYTES IN TM FRAME
TMBYT3  SET      TMBYT1-1H
TMBYT2  EQU      TMBYT3
TMMAX   SET      TMBYT3+30H
        GLOBAL   G43
        GLOBAL   TTYLNK
        GLOBAL   CMTDH
        GLOBAL   RUN
        GLOBAL   DUMP
        SECTION  FLIGHT
RST0    MVI      A,0B4H    ;START REAL TIME CLOCK
        OUT      63H
        JMP      BEGIN     ;JUMP TO CONTINUE
        NOP
RST1    BYTE     0,0,0,0,0,0,0,0
RST2    BYTE     0,0,0,0,0,0,0,0
RST3    BYTE     0,0,0,0,0,0,0,0
RST4    BYTE     0,0,0,0
TRAP    RIM
        JMP      BEGINT    ;THIS IS RF RESET AND REINITIALIZES ALL
                                                   BUT REAL TIME
RST5    BYTE     0,0,0,0
INT5.5  BYTE     0,0,0,0
RST6    BYTE     0,0,0,0
INT6.5  BYTE     0,0,0,0
RST7    BYTE     0,0,0,0
INT7.5  PUSH     PSW       ;****THIS INT OUTPUTS DATA TO TM
        IN       5BH       ;IS THIS THE FIRST FRAME?
        PUSH     H
        PUSH     D
        PUSH     B
        ANI      04H
        JNZ      G9        ;JUMP IF IT IS THE FIRST FRAME
        LDA      802FH     ;GET STATUS WORD AND CHECK FOR BURST
                                                          FLAG
        ANI      10H
        JNZ      G10       ;JUMP IF BURST FLAG IS SET
        LDA      802EH     ;GET TM BYTE COUNT
        CPI      TMBYT1
        JNC      G11       ;JUMP IF THIS IS THE LAST FRAME
        DCR      A         ;USE FRAME COUNTER TO GET NEXT BYTE OUT
        LXI      H,802FH   ;LOAD -HL- WITH POINTER
        MOV      E,A
        MVI      D,00H
        DAD      D         ;ADD POINTER TO FRAME COUNTER
        MOV      A,M       ;GET THE BYTE POINTED TO BY -HL-
        OUT      44H       ;OUTPUT TO TM
        LDA      802EH     ;AGAIN GET TM BYTE COUNT
        INR      A         ;INC BYTE COUNTER
G12     STA      802EH     ;STORE BYTE COUNTER
```

```
G13     POP     B
        POP     D
        POP     H
        POP     PSW
        EI
        RET
G11     LDA     802FH    ;GET STATUS
        OUT     44H      ;OUTPUT TO TM
        ANI     73H      ;REMOVE DATA VALID AND TTY ACTIVE FLAGS
        STA     802FH    ;STORE STATUS
        MVI     A,01H    ;SET FRAME COUNTER TO 1
        JMP     G12
G10     CALL    RECALL   ;DUMPING RAM SO GET BYTE FROM BURST
                                                          BUFFER
        OUT     44H
        JC      G13      ;JUMP IF NOT THE LAST BURST DATA
        MVI     A,00H    ;RESET ALL FLAGS IN STATUS
        STA     802FH    ;STORE STATUS
        JMP     G13
G9      IN      59H      ;RESET FIRST FRAME FLAG
        ANI     0FDH
        OUT     59H
        ORI     02H
        OUT     59H
        MVI     A,02H    ;SET FRAME COUNTER TO 2
        STA     802EH    ;STORE FRAME COUNTER
        LDA     802FH    ;GET STATUS
        OUT     44H
        JMP     G13
FREQC   IN      5AH      ;****CALCULATE FREQ CORRECTION FACTOR
        ANI     02H
        RZ
        PUSH    H
        MOV     A,M      ;GET UPDATE TIME AND ADD 7 TO IT
        INR     A        ;NOW WAIT 7 SECONDS AFTER LEAVING TO
                                                    UPDATA AGAIN
        ANI     07H
        MOV     M,A
        IN      64H      ;GET DATA FROM RF FREQ COUNTER
        CMA
        STA     803CH    ;STORE IN TM FRAME
        OUT     54H      ;OUTPUT TO TOS OF 9511
        IN      64H
        CMA
        STA     803DH    ;STORE IN TM FRAME
        OUT     54H      ;OUTPUT TO TOS OF 9511
        MVI     A,08H    ;FREQ CORR=(F0/F1)**2, F0 IS KNOWN F1
                                               WAS JUST COLLECTED
        OUT     54H
        XRA     A
        OUT     54H
        MVI     A,1CH    ;CONVERT TOS TO FLOATING POINT
```

```
          CALL     WAIT
          XRA      A          ;NOW OUTPUT F0 TO TOS
          OUT      54H
          MVI      A,47H
          OUT      54H
          MVI      A,36H
          OUT      54H
          MVI      A,14H
          OUT      54H
          MVI      A,13H      ;DIVIDE F0 BY F1
          CALL     WAIT
          MVI      A,17H      ;SQUARE F0/F1
          CALL     WAIT
          MVI      A,12H
          CALL     WAIT
          LXI      H,805AH ;GET FREQ CORR AND STORE FOR USE IN
                                                     MAIN PROGRAM
          IN       54H
          MOV      M,A
          INX      H
          IN       54H
          MOV      M,A
          INX      H
          IN       54H
          MOV      M,A
          INX      H
          IN       54H
          MOV      M,A
          IN       59H        ;OUTPUT A START PULSE TO FREQ COUNTER
          ORI      04H
          OUT      59H
          ANI      0FBH
          OUT      59H
          POP      H
          RET
BEGIN     MVI      A,0FFH     ;****INITIALIZATION STARTS HERE IF
                                                 SYSTEM WAS RESET
          OUT      62H        ;LOAD REAL TIME CLOCK WITH 1S
                                                  COMPLIMENT (000
          OUT      62H
          MVI      A,0CEH     ;SET MODE OF USART
          OUT      69H
          MVI      A,27H      ;SET COMMAND OF USART
          OUT      69H
BEGINT    LXI      SP,5900H;****INITILIZATION STARTS HERE IF
                                                           TRAPED
          MVI      A,0C1H     ;SET 8155 TO PA AS OUTPUT AND PB PC AS
                                                           INPUTS
          OUT      58H
          MVI      A,80H      ;SET USARTS CLOCK TO 300 BAUD
          OUT      5CH
          MVI      A,42H
```

```
        OUT     5DH
        MVI     A,74H   ;SET COUNTERO OF 1 TO MODE 1
        OUT     63H
        MVI     A,0FFH  ;SET COUNTERO OF 1 TO 1S COMPLIMENT
                                                        0000
        OUT     61H
        OUT     61H
        MVI     A,34H   ;SET COUNTERO OF 2 TO MODE 1
        OUT     67H
        MVI     A,0FFH  ;SET COUNTERO OF 2 TO 1S COMPLIMENT
                                                        0000
        OUT     64H
        OUT     64H
        MVI     A,0BH   ;TURN ON INTERUPT 7.5
        SIM
        EI
        MVI     A,16H
        OUT     59H     ;INITILIZE PA
        MVI     A,12H
        OUT     59H
        XRA     A       ;RESET WAIT FLAG
        STA     8067H
        MVI     A,80H
        OUT     40H     ;RESET ALL DACS AND TM PORT
        XRA     A
        OUT     44H
G64     OUT     50H
        INR     A
        CPI     20H
        JNZ     G64
        XRA     A
        OUT     50H
        MVI     A,70H   ;SET COUNTER1 OF 2 TO MODE 0
        OUT     67H
        MVI     A,40H   ;RESET USART AND REDEFINE
        OUT     69H
        MVI     A,0CEH
        OUT     69H
        MVI     A,37H
        OUT     69H
        LXI     H,3100H ;SET TMEND AND TMEGN TO FIRST LOCATION
                                                  OF BUFFER
        SHLD    8021H
        SHLD    8023H
G55     INX     H       ;DETERMINE RAM THAT IS AVAILABLE
        MVI     M,00H
        MOV     A,M
        ORA     A
        JNZ     G56
        MVI     M,0FFH
        MOV     A,M
        INR     A
```

```
        JZ      G55
G56     DCX     H
        LXI     D,8100H
        CALL    CMPDH
        JC      G58
        HLT
G58     SHLD    3061H    ;SAVE MAX ADDRESS OF MEMORY
        LXI     H,802EH  ;NOW CLEAR SOME MEMORY
G57     XRA     A
        MOV     M,A
        INX     H
        MOV     A,L
        CPI     46H
        JNZ     G57
        CALL    DUMP     ;NOW DUMP THE TM BUFFER THIS IS DONE TO
                                                    CLEAR BUFFER
        LXI     H,805AH  ;SET THE FREQ CORR FACTOR TO 1
        MVI     M,01H
        INX     H
        MVI     M,80H
        INX     H
        MVI     M,00H
        INX     H
        MVI     M,00H
        JMP     MAIN     ;START THE MAIN PROGRAM
STORE   PUSH    H        ;****STORES DATA INTO TM BUFFER
        LHLD    8021H    ;GET LOCATION OF NEXT BUFFER LOCATION
        MOV     M,A
        INX     H        ;POINT TO NEXT LOCATION
        SHLD    8021H    ;NOW STORE ADDRESS
        POP     H
        RET
TRNSFR  OUT     40H      ;****-ACC- GETS OUTPUTED TO PORT
                                                 DEFINED IN -B-
        MOV     A,B
        OUT     50H
        XRA     A
        OUT     50H
        RET
BIASOT  MOV     A,E      ;****OUTPUTS BIASES TO DACS
        MVI     B,09H    ;B1 IS IN -E-,B2 IN -D-,B3 IN -L-,B4 IN
                                                 -H-,B5 IN -C-
        CALL    TRNSFR   ;-B- LOADED WITH PORT NUMBER OF FIRST
                                                           DAC
        MOV     A,D
        INR     B
        CALL    TRNSFR
        MOV     A,L
        INR     B
        CALL    TRNSFR
        MOV     A,H
        INR     B
```

```
        CALL    TRNSFR
        MOV     A,C
        INR     B
        CALL    TRNSFR
        RET
RECALL  LHLD    8023H   ;****GETS DATA OUT OF TM BUFFER
        XCHG            ;PUT TMEND INTO -HL-,AND TMBGN INTO
                                                        -DE-
        LHLD    8021H
        CALL    CMPDH   ;IS TMEND EQUAL TO TMBGN?
        JNC     G1      ;JUMP IF IT IS
        XCHG
        MOV     A,M     ;GET DATA OUT OF BUFFER
        INX     H
        SHLD    8023H   ;STORE TMBGN
        RET
G1      LXI     H,8100H ;SET TMEND AND TMBGN TO FIRST TM BUFFER
                                                    LOCATION
        SHLD    8021H
        SHLD    8023H
        RET
CMPDH   MOV     A,D     ;****COMPARE -DE- AND -HL-
        CMP     H       ;IF -HL->-DE- CY=1
        RNZ             ;IF -HL-<=-DE- CY=0
        MOV     A,E     ;IF -HL-=-DE- Z=1
        CMP     L
        RET
MAIN    LXI     H,1800H ;****GETS A PAGE FROM BOOK FROM SHELF
        SHLD    8025H
G3      LHLD    8025H
        SHLD    8027H   ;SAVE SHELF POINTER
G2      LHLD    8027H   ;GET SHELF POINTER
        MOV     E,M     ;GET BOOK POINTER
        INX     H
        MOV     D,M
        INX     H
        SHLD    8027H   ;SAVE SHELF POINTER
        XCHG            ;IS THIS THE LAST BOOK OF THIS SHELF?
        LXI     D,0FFFFH
        CALL    CMPDH
        JZ      G3      ;JUMP IF IT IS
G7      MOV     E,M     ;GET PAGE FROM BOOK
        INX     H
        MOV     D,M
        INX     H
        SHLD    8029H   ;SAVE BOOK POINTER
        XCHG
        LXI     D,0FFFFH        ;IS THIS THE LAST PAGE OF THIS
                                                    BOOK?
        CALL    CMPDH
        JZ      G8      ;JUMP IF IT IS
        MVI     A,01H   ;SET FLAG FOR TTYLNK SUB
```

```
        STA     8065H
        CALL    TTYLNK  ;ANY MESSAGES FROM THE GROUND?
        CALL    RUN     ;NOW RUN THIS PAGE
        LHLD    8029H   ;GET BOOK POINTER
        JMP     G7
G8      LHLD    8027H   ;IT WAS LAST PAGE SO GET NEXT BOOK
        MOV     E,M
        INX     H
        MOV     D,M
        INX     H
        SHLD    8027H   ;SAVE BOOK POINTER
        XCHG
        LXI     D,0FFFFH;WAS THIS THE LAST BOOK?
        CALL    CMPDH
        JZ      G2      ;JUMP IF IT WAS AND GET NEXT BOOK
        JMP     G7      ;GO GET NEXT PAGE
DUMP    IN      5AH     ;CHECK TO SEE IF DUMP FLAG IS SET
        ANI     04H
        JZ      G69     ;JUMP IF WE ARE NOT GOING TO DUMP
        LXI     H,805AH
        MVI     M,01H
        INX     H
        MVI     M,80H
        INX     H
        MVI     M,00H
        INX     H
        MVI     M,00H
        JMP     G63
G69     LDA     802FH   ;MOVES TM BUFFER RAM TO TM
        ORI     08H     ;ADD DUMP SOON TO STATUS
        STA     802FH
        MVI     A,CAH   ;SET TIME-OUT TIME TO 10 SEC
        OUT     65H
        XRA     A
        OUT     65H
G14     IN      5BH     ;WAIT FOR TIMER TO TIME-OUT
        ANI     20H
        JZ      G14
        IN      59H     ;SET RAM DUMP REQUEST FLAG
        ORI     01H
        OUT     59H
G15     IN      58H     ;WAIT FOR RAM DUMP ACKNOWLEDGE
        RRC
        RRC
        JNC     G15     ;JUMP IF NOT ACKNOWLEDGED
        MVI     A,0AAH  ;TM OUTPUT TO AA
        OUT     44H
        CALL    FRM16   ;WAIT FOR 16 FRAMES TO GO BY
        MVI     A,10H   ;SET RAM DUMP FLAG IN STATUS
        STA     802FH
G16     LDA     802FH   ;WAIT FOR RAM DUMP TO BE FINISHED
        ORA     A
```

```
        JNZ     G16
        IN      59H      ;RESET DUMP REQUEST
        ANI     0FEH
        OUT     59H
        LXI     H,802FH ;CLEAR TM FRAME
G73     MVI     M,00H    ;WAIT FOR 16 TM FRAMES TO GO BY WITH
                                           STATUS FLAGS RESET
        INX     H
        MOV     A,L
        CPI     3AH
        JNZ     G73
        CALL    FRM16    ;WAIT FOR 16 FRAMES TO PASS
        XRA     A        ;RESET FLAG FOR TTYLNK SUB
        STA     8065H
        CALL    TTYLNK   ;ANY MESSAGES FROM GROUND?
G63     LHLD    8061H    ;GET MAX MEMORY ADDRESS OF TM BUFFER
                                                              RAM
        XCHG
        LXI     H,8100H ;GET STARTING MEMORY ADDRESS OF TM
                                                       BUFFER RAM
G62     MVI     M,00H    ;CLEAR ALL TM BUFFER RAM
        CALL    CMPDH
        INX     H
        JNZ     G62      ;IF NOT FINISHED CONTINUE
        JMP     G1
RUN     LXI     D,8000H ;****RUNS THE PAGE LOCATED IN MEMORY
                                                      -HL- AND UP
        PUSH    H        ;SAVE ADDRESS OF PAGE TO BE USED AS ID
                                                           NUMBER
G17     MOV     A,M      ;MOVE PAGE TO BUFFER STORAGE
        STAX    D
        MOV     A,E
        INX     H
        INX     D
        CPI     15H      ;WAS THAT THE LAST MOVE
        JNZ     G17      ;JUMP IF IT WASNT
        POP     H        ;GET BACK PAGE ID NUMBER
        MOV     A,H
        STAX    D
        INX     D
        MOV     A,L
        STAX    D
G43     LDA     8009H    ;NOW CALCULATE THE NEEDED ROOM IN RAM
                                                    TO STORE PAGE
        ANI     40H      ;ARE BIASES BEING SWEPT?
        JNZ     G19      ;JUMP IF WE ARE
        LHLD    8000H    ;SUBTRACT AMU START FROM AMU END
        MOV     B,H
        MOV     C,L
        LHLD    8002H
        BYTE    08H
        XCHG             ;STORE ANSWER IN -DE-
```

```
        LDA     8015H   ;GET STEPPING INCREAMENT
        ORA     A       ;ARE THERE ANY JUMPING INVOLVED?
        JNZ     G24     ;JUMP IF THERE ISNT ANY
        MVI     A,01H
        STA     8015H
G24     MOV     C,A     ;STORE JUMP FUNCTION IN -BC-
        MVI     B,00H
        LXI     H,0000H ;INITIALIZE -HL- WITH 0000,WE ARE
                                DIVIDING -DE- BY -BC-
        PUSH    H       ;STORE DIVIDEND IN TOS OF STACK
G31     DAD     B       ;NOW DIVIDE
        XTHL
        INX     H
        XTHL
        XCHG
        CALL    CMPDH
        XCHG
        JC      G31     ;JUMP IF NOT FINISHED
        POP     D       ;GET BACK ANSWER
        BYTE    1?H,12H ;MULPIPLY ANSWER BY 4
        MOV     A,E
        ANI     0FCH
        MOV     E,A
        LDA     8009H   ;GET MODE OF PAGE
        ANI     04H     ;ARE WE SWITCHING
        XCHG
        JZ      G53     ;JUMP IF WE ARENT
        DAD     H       ;MULTIPLY BY 2
G53     LDA     8009H   ;ARE WE ACCUMULATING?
        ANI     08H
        JNZ     G21     ;JUMP IF WE ARE
        LDA     8004H   ;MULTIPLY NUMBER OF TIMES BY ANSWER
                                FROM ABOVE
        ORA     A
        RZ              ;RETURN IF NUMBER OF TIMES IS 00
        DCR     A
        JZ      G21     ;JUMP IF ONLY ONE
        MOV     D,H
        MOV     E,L
G61     DAD     D       ;NOW MULTIPLY
        RC              ;RETURN IF NOT ENOUGH MEMORY
        DCR     A
        JNZ     G61     ;CONTINUE IF NOT FINISHED
G21     LXI     D,001CH ;ADD 1C TO TOTAL
        DAD     D
        LXI     D,3F00H ;IF TO BIG DONT RUN PAGE
        CALL    CMPDH
        RC
        PUSH    H       ;SAVE AMOUNT OF MEMORY NEEDED
        LHLD    8021H   ;SUBTRACT TMBGN FROM TMEND
        MOV     C,L
        MOV     B,H
```

```
       LHLD    8061H     ;LOAD MAX TM BUFFER INTO -HL-
       BYTE    09H       ;SUBTRACT
       XCHG              ;-DE- NOW CONTAINS MEMORY AVAILABLE
       POP     H         ;-HL- NOW CONTAINS MEMORY NEEDED
       CALL    CMPDH     ;DO WE HAVE ENOUGH ROOM TO DO PAGE?
       CC      DUMP      ;CALL DUMP IF WE DONT HAVE ENOUGH ROOM
G22    MVI     A,0FAH    ;STORE 24 BIT DUMP SYNC
       CALL    STORE
       MVI     A,0F3H
       CALL    STORE
       MVI     A,20H
       CALL    STORE
       MVI     A,80H     ;LATCH IN REAL TIME AND STORE
       OUT     63H
       IN      62H
       CMA
       MOV     B,A
       IN      62H
       CMA
       CALL    STORE
       MOV     A,B
       CALL    STORE
       LXI     H,8000H   ;MOVE PAGE INTO TM BUFFER RAM
G23    MOV     A,M
       CALL    STORE
       MOV     A,L
       CPI     15H       ;LAST ONE TO STORE?
       INX     H
       JNZ     G23       ;JUMP IF ITS NOT
       LDA     8004H     ;GET NUMBER OF LOOP COUNTER
       STA     8059H     ;STORE FOR USE LATER
       LHLD    8005H     ;GET DWELL TIME
       LXI     D,0001H   ;DWELL TIME MUST BE LARGER THAN 0002
       CALL    CMPDH
       JC      G65
       LXI     H,0002H
       SHLD    8005H     ;STORE NEW DWELL TIME AND OUTPUT TO
                                                  DWELL COUNTER
G65    MVI     A,32H
       OUT     63H
       SHLD    6000H
       LHLD    800AH     ;GET BIASES THEN OUTPUT TO DACS
       XCHG
       LHLD    800CH
       LDA     800EH
       MOV     C,A
       CALL    BIASOT
       LHLD    8007H     ;GET RATIO AND FLAGS AND OUTPUT TO DACS
                                                      AND FLAGS
       MOV     A,L
       MVI     B,01H
       CALL    TRNSFR
```

```
        LDA     8009H
        ANI     10H
        RRC
        ORA     B
        INR     B
        CALL    TRNSFR
        MVI     A,01H
        STA     8066H
        LDA     8009H   ;WHICH TYPE OF SWEEP?
        RLC
        RLC
        JC      G25     ;JUMP IF BIAS SWEEP
        RLC
        CC      AMU     ;CALL IF AMU SWEEP
G26     MVI     A,80H   ;GET AND STORE REAL TIME
        OUT     63H
        IN      62H
        MOV     B,A
        IN      62H
        CMA
        CALL    STORE
        MOV     A,B
        CMA
        CALL    STORE
        MVI     B,07H   ;RESET AMU TO 000
        XRA     A
        CALL    TRNSFR
        INR     B
        CALL    TRNSFR
        RET
G25     CALL    BIAS
        JMP     G26
G19     LXI     H,800AH ;GO THROUGH ALL BIASES AND FIND LONGEST
                                                           SWEEP
        LXI     D,800FH
        MVI     B,00H
G47     LDAX    D       ;GET PRIMARY BIAS
        SUB     M       ;SUBTRACT FROM SECONDARY SWEEP BIAS
        CMP     B       ;IS THIS ONE LARGER THAN THE LAST ONE?
        INX     H
        INX     D
        JC      G44     ;JUMP IF IT IS
        MOV     B,A     ;MOVE NEW MAX INTO MAX BUFFER
G44     MOV     A,L     ;WAS THIS THE LAST ONE?
        CPI     0FH
        JNZ     G47     ;JUMP IF IT ISNT
        MOV     E,B     ;MOVE MAX INTO BUFFER FOR MEMORY NEEDED
                                                      CALCULATION
        MVI     D,00H
        BYTE    18H,1CH ;MULTIPLY BY 4
        MOV     A,E
        ANI     0FCH
```

```
        MOV     E,A
        XCHG
        JMP     G53
WAIT    OUT     55H      ;****OUTPUTS COMMAND TO 9511 AND WAITS
                                            FOR IT TO FINISH
G27     IN      55H
        RLC
        JC      G27
        RET
COLECT  MVI     A,40H    ;****COLLECTS,CORRECTS,AND STORES DATA
        OUT     63H      ;LATCH IN NEW DATA
        IN      61H      ;LOAD DATA INTO -HL-
        CMA
        MOV     L,A
        IN      61H
        CMA
        MOV     H,A
        CALL    CORDAT   ;CORRECT THE DATA
        LDA     8009H    ;DOES THIS PAGE DOWN COUNT?
        RLC
        JC      G28      ;JUMP IF WE DONT DOWN COUNT
        LXI     D,0800H  ;IS DATA GREATER THAN 800H?
        CALL    CMPDH
        JC      G28      ;JUMP IF IT IS AND DONT DOWN COUNT
        IN      5AH      ;IS NO DOWN FLAG SET?
        RRC
        JC      G28      ;JUMP IF IT IS AND DONT DOWN COUNT
        XRA     A        ;RESET DOWN COUNT FLAG
        STA     802DH
        XCHG
        LDA     8066H    ;CHECK TO SEE IF WE ARE TO UPDATE NOISE
                                                          COUNT
        DCR     A
        JNZ     G5       ;JUMP IF WE ARE NOT GOING TO UPDATE
        MVI     A,05H    ;UPDATE NOISE COUNT
        STA     8066H
        MVI     A,0FFH   ;SET RATIO TO 3FF AND COLLECT
                                            BACKGROUND NOISE
        MVI     B,01H
        CALL    TRNSFR
        INR     B
        LDA     8009H
        ANI     10H      ;SAVE PRESENT FLAG STATUS
        ORI     03H
        CALL    TRNSFR
        MVI     A,74H    ;RESET DATA COUNTER
        OUT     63H
        MVI     A,0FFH
        OUT     61H
        OUT     61H
        CALL    STZERO   ;CORRECT DATA COLLECTION SIGNALS
        MVI     A,80H    ;OUTPUT A START OF COLLECTION PULSE
```

```
        OUT     50H
        XRA     A
        OUT     5CH
        CALL    DATCOL   ;WAIT FOR DATA TO FINISH COLLECTING
        MVI     A,40H    ;LATCH IN BACKGROUND NOISE DATA
        OUT     63H
        IN      61H      ;GET NOISE COUNT AND STORE IN -HL-
        CMA
        MOV     L,A
        IN      61H
        CMA
        MOV     H,A
        PUSH    D        ;SAVE DATA COUNT
        CALL    CORDAT   ;CORRECT NOISE COUNT
        POP     D        ;GET BACK DATA COUNT
        PUSH    H
        LHLD    8007H
        MOV     A,L
        MVI     B,01H
        CALL    TRNSFR
        LDA     8009H
        ANI     10H
        RRC
        ORA     B
        INR     B
        CALL    TRNSFR
        POP     H
        MOV     A,H      ;SUBTRACT
        CMA
        MOV     H,A
        MOV     A,L
        CMA
        MOV     L,A
        INX     H
        SHLD    8044H    ;SAVE BACKGROUND NOISE COUNT
G67     DAD     D
        PUSH    H        ;SAVE CORRECTED DATA
        MOV     A,H      ;CHECK TO SEE IF DATA IS NEGATIVE
        RLC
        JNC     G30      ;JUMP IF DATA IS POSITIVE
        MOV     A,H
        CMA              ;NOW COMPLIMENT THE DATA AND SET MSB
        ORI     80H
        MOV     H,A
        MOV     A,L
        CMA
        MOV     L,A
        POP     D        ;REPLACE OLD DATA WITH UPDATED DATA
        PUSH    H
G30     LDA     8009H    ;GET MODE AND CHECK FOR ACCUMULATION
        ANI     08H
        JZ      G18      ;JUMP IF NO ACCUMULATION
```

```
        LHLD    801CH   ;GET PRESENT AMU AND STORE IN TM BUFFER
                                                           RAM
        MOV     A,H
        CALL    STORE
        MOV     A,L
        CALL    STORE
        LHLD    8021H   ;GET TREND POINTER AND GET PAST
                                           ACCUMULATED DATA
        MOV     D,M
        INX     H
        MOV     E,M
        POP     H       ;GET COLLECTED AND CORRECTED DATA
        CALL    ADDDAT  ;ADD THE TWO TOGETHER
        MOV     A,H
        CALL    STORE
        MOV     A,L
        CALL    STORE
        XCHG
G35     LXI     H,802FH ;NOW STORE PARAMETERS TO BE SENT
                                        THROUGH TM TO GROUND
        LDA     802DH   ;LOAD -ACC- WITH NO DOWN FLAG
        ANI     40H
        ORI     80H     ;SET DATA READY FLAG
        MOV     B,A     ;SAVE FLAGS
        IN      69H
        ANI     80H
        RRC
        RRC
        ORA     B
        MOV     B,A
        IN      5AH     ;GET HV MONITOR
        ANI     08H
        RRC
        RRC
        ORA     B
        MOV     B,A
        CALL    FRM1
        MOV     M,B     ;STORE STATUS IN TM FRAME
        INX     H
        LDA     8016H   ;STORE ID IN TM FRAME
        MOV     M,A
        INX     H
        LDA     8017H
        MOV     M,A
        INX     H
        MOV     M,D     ;STORE DATA IN TM FRAME
        INX     H
        MOV     M,E
        INX     H
        LDA     8006H   ;STORE DWELL TIME IN TM FRAME
        MOV     M,A
        INX     H
```

```
        LDA     8005H
        MOV     M,A
        INX     H
        XCHG            ;GET RATIO AND ROTATE TO RIGHT 2 TIMES
        LHLD    8067H
        BYTE    10H,10H
        XCHG
        MOV     M,E
        INX     H
        INX     H
        XCHG
        LHLD    801CH   ;GET AMU AND ROTATE TO THE LEFT 4 TIMES
        XCHG
        BYTE    18H,18H,18H,18H
        MOV     A,E
        ANI     0F0H
        MOV     M,D     ;STORE AMU IN TM FRAME
        INX     H
        MOV     M,A
        INX     H
        LDA     8009H   ;STORE MODE IN TM FRAME
        MOV     M,A
        INX     H
        LDA     8004H   ;STORE LOOP COUNT IN TM FRAME
        MOV     M,A
        RET
G18     LHLD    801CH   ;GET PRESENT AMU
        MOV     A,H
        CALL    STORE   ;STORE AMU IN TM BUFFER RAM
        MOV     A,L
        CALL    STORE
        POP     D       ;GET BACK DATA
        MOV     A,D
        CALL    STORE
        MOV     A,E
        CALL    STORE
        JMP     G35
G5      LHLD    8044H   ;GET OLD BACKGROUND NOISE COUNT
        STA     8066H
        JMP     G67
DATCOL  IN      58H     ;****WAIT FOR DATA TO BE COLLECTED
        ANI     08H
        JNZ     DATCOL  ;JUMP IF COLLECTION HASN'T STARTED
G29     IN      58H
        ANI     08H
        JZ      G29     ;JUMP IF COLLECTION ISN'T FINISHED
        RET
AMU     LHLD    8021H   ;****RUNS AN AMU SWEEP PAGE,GET TMEEP
                                AND STORE FOR LATER
        SHLD    8057H
        LHLD    8000H   ;GET FIRST AMU AND MULTIPY BY FRQ
                                CORRECTION FACTOR
```

```
        MOV     A,L
        OUT     54H         ;OUTPUT AMU TO TOS OF 9511
        MOV     A,H
        OUT     54H
        SHLD    8023H       ;STORE AMU START IN CASE OF
                                          ACCUMULATION MODE
        MVI     A,1DH       ;CONVERT TOS TO FLOATING POINT
        CALL    WAIT
        LXI     H,805DH     ;PUT FREQ CORRECTION FACTOR ONTO TOS OF
                                                        9511
        MOV     A,M
        OUT     54H
        DCX     H
        MOV     A,M
        OUT     54H
        DCX     H
        MOV     A,M
        OUT     54H
        DCX     H
        MOV     A,M
        OUT     54H
        MVI     A,12H       ;MULTIPLY TOS BY NOS
        CALL    WAIT
        MVI     A,1FH       ;CONVERT TOS TO FIXED POINT
        CALL    WAIT
        IN      54H         ;LOAD -HL- WITH CORRECTED AMU
        MOV     L,A
        IN      54H
        MOV     H,A
G37     MOV     A,H         ;TRANSFER AMU TO AMU DACS
        MVI     B,07H
        CALL    TRNSFR
        INR     B
        MOV     A,L
        CALL    TRNSFR
        MVI     A,74H
        OUT     63H
        MVI     A,0FFH      ;RESET DATA COUNTER
        OUT     61H
        OUT     61H
        CALL    STZERO      ;CORRECT DATA INPUT LEVEL
        MVI     A,30H       ;OUTPUT A START OF COLLECTION PULSE
        OUT     50H
        XRA     A
        OUT     50H
        LHLD    8000H       ;GET AMU AND ADJUST TO DO NEXT STEP
        SHLD    801CH       ;STORE AMU BECAUSE THIS IS PRESENT AMU
        LDA     8015H       ;GET STEP VALUE
        MOV     E,A
        MVI     D,00H       ;ADD STEP TO AMU
        DAD     D
G6?     SHLD    8000H       ;STORE NEW AMU
```

```
MOV     A,L       ;CORRECT AMU WITH FREQ CORRECTION
                                                FACTOR
OUT     54H
MOV     A,H
OUT     54H
MVI     A,1DH     ;CONVERT TOS TO FLOATING POINT
CALL    WAIT
LXI     H,805DH   ;OUTPUT FREQ CORRECTION FACTOR TO TOS
MOV     A,M
OUT     54H
DCX     H
MOV     A,M
OUT     54H
DCX     H
MOV     A,M
OUT     54H
DCX     H
MOV     A,M
OUT     54H
MVI     A,12H     ;MULTIPLY TOS BY NOS
CALL    WAIT
MVI     A,1FH     ;CONVERT TOS TO FIXED POINT
CALL    WAIT
IN      54H       ;GET CORRECTED AMU
MOV     L,A
IN      54H
MOV     H,A
SHLD    8063H     ;STORE CORRECTED AMU
MVI     A,80H     ;IS IT TIME TO UPDATE FREQ CORRECTION
                                                FACTOR?
OUT     63H
IN      62H
ANI     07H
LXI     H,8020H   ;CHECK AGAINST STORED TIME
CMP     M
IN      62H
CZ      FREQC     ;CALL IF IT IS TIME
CALL    DATCOL    ;WAIT FOR DATA TO BE COLLECTED
CALL    COLECT    ;GET DATA
LDA     8009H     ;GET MODE OF PAGE
ANI     04H       ;ARE WE SWITCHING BIASES?
JZ      G39       ;JUMP IF WE ARENT
LHLD    800FH     ;GET ALL SECONDARY BIASES
XCHG
LHLD    8011H
LDA     8013H
MOV     C,A
MVI     A,74H
OUT     63H
MVI     A,0FFH
OUT     61H
OUT     61H
```

```
        CALL    BIASOT  ;OUTPUT ALL BIASES TO THERE DACS
        CALL    STZERO  ;CORRECT DATA INPUT LEVEL
        MVI     A,80H   ;OUTPUT A START OF COLLECTION PULSE
        OUT     50H
        XRA     A
        OUT     50H
        CALL    DATCOL  ;WAIT FOR DATA TO FINISH COLLECTING
        CALL    COLECT  ;GET DATA
G39     LHLD    801CH   ;GET PRESENT AMU
        XCHG
        LHLD    8002H   ;GET FINAL AMU
        CALL    CMPDH   ;IS THIS THE LAST AMU TO BE SCANNED?
        JC      G40     ;JUMP IF IT ISNT
        LDA     8004H   ;GET NUMBER OF TIMES
        DCR     A       ;DECREAMENT NUMBER OF TIMES
        STA     8004H
        JNZ     G41     ;JUMP IF IT WASNT THE LAST LOOP
ENDING  CALL    FIN1    ;****ENDS A PAGE, ALSO GETS COUNTS PER
                                                            SECOND
        MVI     A,00H
        STA     802FH
        LHLD    8005H   ;GET DWELL TIME OF PAGE
        XRA     A       ;OUTPUT A 100 TO TOS OF 9511
        OUT     54H
        OUT     54H
        MVI     A,0C8H
        OUT     54H
        MVI     A,08H
        OUT     54H
        MOV     A,L     ;NOW OUTPUT DWELL TIME TO TOS
        OUT     54H
        MOV     A,H
        OUT     54H
        MVI     A,1DH   ;CONVERT TOS TO FLOATING POINT
        CALL    WAIT
        MVI     A,13H   ;DIVIDE NOS BY TOS
        CALL    WAIT
        MVI     A,17H   ;DUPLICATE TOS TO NOS
        CALL    WAIT
        LHLD    8057H   ;POINT TO FIRST DATA TO BE DIVIDED
        INX     H
        INX     H
G42     MOV     D,M     ;GET DATA
        INX     H
        MOV     E,M
        DCX     H
        XCHG
        MOV     A,L
        RAL             ;IS DATA NEGATIVE?
        CC      NEG     ;CALL IF IT IS NEGATIVE AND CORRECT
                                                            DATA
        MOV     A,L     ;PUT DATA INTO TOS OF 9511
```

```
        OUT     54H
        MOV     A,L
        OUT     54H
        MVI     A,1DH       ;CONVERT TOS TO FLOATING POINT
        CALL    WAIT
        MVI     A,12H       ;MULTIPY TOS BY NOS
        CALL    WAIT
        LDA     8009H       ;GET MODE OF PAGE
        ANI     08H         ;ARE WE ACCUMULATING?
        JZ      G38         ;JUMP IF WE ARENT
        LDA     8059H       ;GET NUMBER OF TIMES AND OUTPUT TO TOS
        OUT     54H
        XRA     A
        OUT     54H
        MVI     A,1DH       ;CONVERT TOS TO FLOATING POINT
        CALL    WAIT
        MVI     A,13H       ;DIVIDE NOS BY TOS
        CALL    WAIT
G38     IN      54H         ;GET COUNTS PER SECOND DATA
        MOV     H,A
        IN      54H
        MOV     L,A
        IN      54H
        RLC
        IN      54H
        MOV     A,L
        RAL
        MOV     L,A
        MVI     A,17H       ;DUPLICATE TOS TO NOS
        CALL    WAIT
        XCHG                ;GET BACK POINTER
        MOV     M,D
        INX     H
        MOV     M,E
        INX     H
        INX     H
        INX     H
        XCHG
        LHLD    8021H       ;GET TMEND POINTER
        CALL    CMPDH       ;LAST DATA TO CONVERT?
        XCHG
        JC      G42         ;JUMP IF THERE ARE MORE
        RET
G40     LHLD    800AH       ;SWITCH BIASES BACK TO NORMAL
        XCHG
        LHLD    800CH
        LDA     800EH
        MOV     C,A
        CALL    BIASOT
        LHLD    8063H       ;GET CORRECTED AMU
        JMP     G37
G41     LHLD    802BH       ;GET ORIGINAL AMU AND PUT IT IN PRESENT
```

```
                                                      AMU LOCATION
        SHLD    8000H
        LDA     8009H     ;GET MODE OF PAGE
        ANI     08H       ;ARE WE ACCUMULATING?
        JZ      AMU       ;JUMP IF WE ARENT
        LHLD    8057H     ;SET TM BUFFER RAM POINTER TO CORRECT
                                                          LOCATION
        SHLD    8021H
        JMP     AMU
G28     MVI     A,0FFH
        STA     802DH
        PUSH    H
        JMP     G30
NEG     CMC               ;****IF DATA IS NEGATIVE THEN TELL 9511
        RAR               ;AND RESET MSBIT OF DATA
        MOV     H,A
        MVI     A,15H     ;CHANGE SIGN IN 9511 TOS
        CALL    WAIT
        RET
BIAS    LHLD    8021H     ;****SWEEP THROUGH BIASES WITH AMU
                                                          CONSTANT
        SHLD    8057H     ;STORE TMEND IN CASE WE ACCUMULATE
        LHLD    8000H     ;GET AMU
        SHLD    801CH     ;STORE AMU FOR TM FRAME STORAGE
        LDA     8014H
        ANI     1FH
        STA     8014H
        MOV     A,L       ;CORRECT AMU WITH FREQUENCY CORRECTION
                                                            FACTOR
        OUT     54H
        MOV     A,H
        OUT     54H
        MVI     A,1DH     ;CONVERT TOS TO FLOATING POINT
        CALL    WAIT
        LXI     H,805DH   ;POINT TO FREQUENCY CORRECTION FACTOR
        MOV     A,M
        OUT     54H
        DCX     H
        MOV     A,M
        OUT     54H
        DCX     H
        MOV     A,M
        OUT     54H
        DCX     H
        MOV     A,M
        OUT     54H
        MVI     A,12H     ;MULTIPLY TOS BY NOS
        CALL    WAIT
        MVI     A,1FH     ;CONVERT TOS TO FIXED POINT
        CALL    WAIT
        IN      54H       ;GET CORRECTED AMU
        MOV     L,A
```

```
        IN      54H
        MVI     B,07H    ;NOW OUTPUT AMU TO DACS
        CALL    TRNSFR
        INR     B
        MOV     A,L
        CALL    TRNSFR
        LXI     H,800AH  ;MOVE PRIMARY BIASES INTO BUFFER FOR
                                                        USE LATER
        LXI     D,803FH
G45     MOV     A,M
        STAX    D
        INX     D
        INX     H
        MOV     A,L
        CPI     0FH
        JNZ     G45      ;JUMP IF MORE ARE TO BE MOVED
G46     CALL    STZERO   ;CORRECT INPUT DATA LEVEL
        MVI     A,80H    ;OUTPUT A START OF COLLECTION PULSE
        OUT     50H
        XRA     A
        OUT     50H
        CALL    DATCOL   ;WAIT FOR DATA TO BE COLLECTED
        CALL    COLECT   ;GET DATA
        LXI     H,8009H  ;FIND OUT WHICH BIASES ARE TO BE
                                                     INCREAMENTED
        LDA     8014H    ;GET BIAS MASK
        ORA     A
        JZ      G48A
G48     RRC              ;CHECK EACH BIT FOR A ONE
        INX     H
        JNC     G48      ;JUMP IF NOT TO BE INCREAMENTED
        MOV     A,M      ;SAVE THIS BIAS
        LXI     D,0005H
        DAD     D        ;ADD 5 TO POINTER TO GET ENDING BIAS
        CMP     M        ;ARE THERE ANY MORE TO BE INCREAMENTED
        JNZ     G49      ;JUMP IF IT WASNT
G48A    LDA     8004H    ;IS THIS THE LAST LOOP
        DCR     A
        STA     8004H
        JZ      ENDING   ;JUMP IF THIS IS THE END
        LXI     H,800AH  ;SET UP TO MAKE ANOTHER LOOP
        LXI     D,803FH
G51     LDAX    D        ;MOVE ORIGINAL BIASES INTO BIAS
                                                        LOCATIONS
        MOV     M,A
        INX     D
        INX     H
        MOV     A,L
        CPI     0FH
        JNZ     G51      ;JUMP IF NOT DONE MOVING
        LDA     8009H    ;GET MODE OF PAGE
        ANI     07H      ;ARE WE ACCUMULATING?
```

```
        JZ      G46       ;JUMP IF WE ARENT
        LHLD    8057H     ;SET BURST BUFFER TO CORRECT LOCATION
        SHLD    8021H
        JMP     G46
G49     LDA     8014H     ;GET BIAS INCREAMENT MASK
        LXI     H,800AH   ;POINT TO FIRST BIAS
G52     RRC               ;IS THIS BIAS SUPPOSED TO BE
                                                    INCREAMENTED?
        MOV     B,A
        JNC     G54       ;JUMP IF IT ISNT
        INR     M
G54     INX     H         ;POINT TO NEXT BIAS LOCATION
        MOV     A,L
        CPI     0FH       ;LAST BIAS?
        MOV     A,B
        JNZ     G52       ;JUMP IF THERE ARE MORE
        LHLD    800AH     ;GET ALL BIASES AND OUTPUT TO DACS
        XCHG
        LHLD    800CH
        LDA     800EH
        MOV     C,A
        CALL    BIASOT
        JMP     G46
ADDDAT  MOV     A,H       ;****ADDS NEW DATA TO ACCUMULATED DATA
        RLC
        JC      G33       ;JUMP IF ONE DATA IS NEGATIVE
        MOV     A,D
        RLC
        JC      G34       ;JUMP IF NEGATIVE
        DAD     D         ;ADD TOGETHER IF BOTH ARE POSITIVE
        RET
G33     MOV     A,D       ;CHECK FOR THE OTHER DATA TO BE
                                                    NEGATIVE
        RLC
        JC      G58       ;JUMP IF BOTH ARE NEGATIVE
G56     MOV     A,H       ;SUBTRACT ONE FROM THE OTHER
        RAL
        CMC               ;RESET MSB OF NEGATIVE NUMBER
        RAR
        CMA
        MOV     H,A
        MOV     A,L
        CMA
        MOV     L,A
        INX     H
        DAD     D         ;ADD TOGETHER
        MOV     A,H
        RAL
        RNC               ;RETURN IF POSITIVE DIFFERENCE
        CMC
        RAR
        CMA               ;COMPLIMENT DATA AND SET MSB
```

```
        MOV     H,A
        MOV     A,L
        CMA
        MOV     L,A
        RET
G58     DAD     D       ;BOTH NEGATIVE SO ADD AND SET MSB
        MOV     A,H
        ORI     80H
        MOV     H,A
        RET
G34     XCHG
        JMP     G58
STZERO  IN      5BH     ;SET THE INPUT TO DATA COUNTER TO ZERO
        ANI     01H
        JZ      G72     ;JUMP IF IT IS 0
        IN      59H     ;SET TO ZERO
        XRI     08H
        OUT     59H
G72     IN      59H     ;NOW RESET LSB CYCLE DETECTOR
        ANI     0EFH
        OUT     59H
        ORI     10H
        OUT     59H
        RET
CORPAT  IN      5BH     ;TWO PULSES GONE BY?
        ANI     10H
        JNZ     G74
        LXI     H,0000H
        JMP     G71
G74     LXI     D,0000H ;CORRECT THE DATA COLLECTED
        CALL    CMPDH   ;IS DATA ZERO?
        JNZ     G70     ;JUMP IF ITS NOT
        INX     H
        INX     H
G71     IN      5BH     ;IS THE DATA STILL HI?
        ANI     01H
        RZ
        INX     H
        RET
G70     XCHG            ;IF DATA ISNT ZERO THEN SHIFT LEFT
        BYTE    29H
        MOV     A,E
        ANI     0FEH
        MOV     E,A
        INX     D       ;THEN ADD 2 TO IT
        INX     D
        XCHG
        JMP     G71
        END
Z
```

```
        $TITLE   "FLIT2 PROGRAMS FOR BBINS WRITTEN BY JIM
                                                          MANLEY"
        GLOBAL   TMBYT2,FRM1,FRM16
        GLOBAL   G43
        GLOBAL   DUMP
        GLOBAL   TTYLNK
        GLOBAL   CMPDH
        GLOBAL   RUN
TTYLNK  IN       69H       ;COMMUNICATIONS LINK FOR BALLOON
        RAL                ;CHECK DSR
        RNC                ;RETURN IF LINK IS BROKEN
        PUSH     H         ;SAVE -HL- FOR RETURN
G1      MVI      A,02H     ;SET THE TIMER TO TIME-OUT IN 2 SECONDS
        OUT      65H
        XRA      A
        OUT      65H
        IN       69H
        ANI      02H
        JZ       G33
        IN       68H       ;CLEAR USART
G33     MVI      A,05H     ;SEND ENQ TO GROUND STATION
        CALL     TRNSMT
G31     LXI      H,5800H ;START LOADING DATA FROM GROUND INTO 3
                                                          BUFFERS
        CALL     RECEV
        JC       G3
        MOV      B,A
        MVI      A,10H
        OUT      65H       ;SET THE TIMER TO TIME OUT IN 16
                                                          SECONDS
        XRA      A
        OUT      65H
        MOV      A,B
        JMP      G37
G2      CALL     RECEV
        JC       G3        ;JUMP IF ERROR IN SUB
G37     CPI      03H       ;CHECK FOR ETX
        MOV      M,A
        INX      H         ;POINT TO NEXT BUFFER LOCATION
        JZ       G4        ;JUMP IF IT WAS ETX
        MOV      A,L       ;CHECK -HL- FOR END OF BUFFER
        CPI      43H
        JNZ      G2        ;JUMP AND GET ANOTHER CHARECTOR
G4      LXI      H,5843H ;POINT TO BEGINNING OF SECOND BUFFER
G5      CALL     RECEV     ;GET DATA
        JC       G3        ;JUMP IF ERROR IN SUB
        CPI      03H       ;CHECK FOR ETX
        MOV      M,A
        INX      H         ;POINT TO NEXT BUFFER LOCATION
        JZ       G6        ;JUMP IF IT WAS ETX
        MOV      A,L       ;CHECK -HL- FOR END OF BUFFER
        CPI      86H
```

```
        JNZ     G5       ;GET NEXT CHARECTOR
G6      LXI     H,5886H  ;POINT TO BEGINNING OF THIRD BUFFER
G7      CALL    RECEV    ;GET DATA
        JC      G3       ;JUMP IF ERROR IN SUB
        CPI     03H      ;CHECK FOR ETX
        MOV     M,A
        INX     H        ;POINT TO NEXT BUFFER LOCATION
        JZ      G8       ;JUMP IF IT WAS AN ETX
        MOV     A,L      ;CHECK -HL- FOR END OF BUFFER
        CPI     0C8H
        JNZ     G7       ;JUMP IF IT WASNT THE END
G8      LXI     H,5800H  ;LOAD -HL- -DE- -BC- WITH START OF EACH
                                                          BUFFER
        LXI     D,5843H
        LXI     B,5886H  ;NOW COMPARE AND FIND ONE COMPLETE
                                                         MESSAGE
G9      LDAX    D        ;GET DATA FROM SECOND BUFFER
        CMP     M        ;COMPARE AGAINST DATA FROM SECOND
                                                          BUFFER
        JNZ     G10      ;JUMP IF NOT EQUAL
G11     INX     H        ;POINT TO NEXT DATA BYTE
        INX     D
        INX     B
        CPI     03H      ;WAS THIS LAST DATA BYTE AN ETX?
        JZ      G12      ;JUMP IF IT WAS
        MOV     A,L      ;CHECK TO SEE IF THIS WAS THE END OF
                                                     THE BUFFERS
        CPI     42H
        JNZ     G9       ;JUMP IF MORE TO BE COMPARED
G12     LXI     H,5800H  ;POINT TO BEGINNING OF COMPLETE LIST
        MOV     A,M      ;GET FIRST BYTE IS IT A STX?
        CPI     02H
        JNZ     G13      ;JUMP IF NOT AND TRANSMIT A REPEAT
        LDA     5801H    ;SEE IF COMMAND IS A VALID ONE
        CPI     01H
        JZ      G32
        CPI     02H
        JZ      G32
        CPI     04H
        JZ      G32
        CPI     08H
        JZ      G32
        CPI     10H
        JZ      G32
        CPI     20H
        JZ      G32
        CPI     40H
        JNZ     G13
G32     MOV     A,M
G29     CALL    TRNSMT   ;NOW TRANSMIT TOTAL LIST
        INX     H
        CPI     03H      ;CHECK FOR ETX
```

```
        JZ      G30
        MOV     A,L      ;CHECK -HL- FOR MAX BUFFER
        CPI     43H
        MOV     A,M
        JNZ     G29
G30     LXI     H,5301H ;LOAD -HL- WITH SECOND DATA BYTE
                                                  LOCATION
        LDA     8065H    ;IS THIS AFTER RAM DUMP?
        ORA     A
        MOV     A,M      ;GET COMMAND BYTE
        JZ      G20      ;BYPASS RUNNING PROGRAMS IF LOOKING FOR
                                              DUMP FUNCTIONS
        CPI     01H      ;IS COMMAND A NEW PAGE COMMAND?
        JZ      G14      ;JUMP IF IT IS
        CPI     02H      ;IS IT A RUN BOOK COMMAND
        JZ      G15
        CPI     04H      ;IS IT A RUN PAGE COMMAND?
        JZ      G18      ;JUMP IF IT IS
        CPI     10H      ;IS IT A GOTO COMMAND?
        JZ      G28
G20     CPI     08H      ;IS IT A DUMP COMMAND?
        JZ      G19      ;JUMP IF IT IS
        CPI     20H      ;IS IT A WAIT COMMAND?
        JZ      G34      ;JUMP IF IT IS
        CPI     40H      ;IS IT A CONTINUE COMMAND?
        JZ      G39      ;JUMP IF IT IS
        CPI     03H      ;IS IT AN ETX?
        JZ      G3       ;JUMP IF IT IS
G13     MVI     A,15H    ;SEND A REPEAT CHARECTOR
        CALL    TRNSMT
        JMP     G31
G28     LDA     58CEH    ;CHECK LAST BYTE FOR ETX
        CPI     03H
        JNZ     G3       ;JUMP IF ERROR
        LXI     D,58CDH ;POINT TO END OF DATA
        INX     H        ;POINT TO BEGINNING OF DATA
        CALL    CNVRT    ;CONVERT ASCII TO HEX
        LXI     H,5802H ;NOW GET REPERTOIRE
        MOV     D,M
        INX     H
        MOV     E,M
        INX     H
        XCHG
        SHLD    8027H    ;STORE IN SHELF POINTER
        SHLD    8025H
        XCHG
        MOV     D,M      ;NOW GET PROGRAM
        INX     H
        MOV     E,M
        INX     H
        XCHG
        SHLD    8029H    ;STORE IN PAGE POINTER
```

```
        XCHG
        MOV     D,M
        INX     H
        MOV     E,M
        XCHG
        POP     D
        PUSH    H
G2      XRA     A       ;CLEAR WAIT FLAG
        STA     8067H
G3      LDA     8067H
        RLC
        JC      G1
        POP     H
        RET
G34     MVI     A,0FFH
        STA     8067H
        CALL    FRM16
        JMP     G1
G10     LDAX    B       ;GET DATA FROM THIRD BUFFER
        CMP     M       ;COMPARE TO FIRST BUFFER DATA
        JZ      G11     ;JUMP IF THEY ARE EQUAL
        XCHG            ;PUT -DE- INTO -HL-
        CMP     M       ;COMPARE THIRD BUFFER DATA WITH SECOND
                                                  BUFFER DATA
        JNZ     G13     ;NO COMPARISON GO TRANSMIT A REPEAT
        XCHG
        MOV     M,A     ;LOAD GOOD DATA INTO FIRST BUFFER
        JMP     G11
G14     LDA     5832H   ;CHECK LAST DATA BYTE FOR ETX
        CPI     03H
        JNZ     G3      ;JUMP IF NOT AN ETX CHARECTOR
        INX     H       ;POINT TO NEXT BYTE
        LXI     D,5831H ;LOAD -DE- WITH LAST BYTE OF DATA
        CALL    CNVRT   ;CONVERT AND COMBINE DATA
        LXI     H,5802H ;POINT TO BEGINNING OF DATA
        LXI     D,8000H ;POINT TO BEGINNING OF DESTINATION
        MOV     B,M
        INX     H
        MOV     A,M
        INX     H
        STAX    D
        MOV     A,B
        INX     D
        STAX    D
        INX     D
        MOV     B,M
        INX     H
        MOV     A,M
        INX     H
        STAX    D
        MOV     A,B
        INX     D
```

```
        STAX    D
        INX     D
        MOV     A,M
        INX     H
        STAX    D
        INX     D
        MOV     B,M
        INX     H
        MOV     A,M
        INX     H
        STAX    D
        MOV     A,B
        INX     D
        STAX    D
        INX     D
        MOV     B,M
        INX     H
        MOV     A,M
        INX     H
        STAX    D
        MOV     A,B
        INX     D
        STAX    D
        INX     D
G40     MOV     A,M
        INX     H
        STAX    D
        INX     D
        MOV     A,E
        CPI     16H
        JNZ     G40
        MOV     A,M
        STAX    D
        INX     H
        MOV     A,M
        INX     D
        STAX    D
        XRA     A
        STA     8067H
        CALL    G43
        POP     H
        RET
G19     XRA     A
        STA     8067H
        CALL    DUMP
        POP     H
        RET
G18     LDA     5806H   ;LAST BYTE SHOULD BE A ETX
        CPI     03H
        JNZ     G2      ;JUMP IF ITS NOT
        LXI     D,5805H ;LOAD -DE- WITH LAST LOCATION
        INX     H
```

```
         CALL    CNVRT
         CALL    PACK
         XRA     A
         STA     8067H
         CALL    RUN
         POP     H
         RET
G15      LDA     5806H      ;CHECK LAST BYTE FOR ETX
         CPI     03H
         JNZ     G3         ;JUMP IF NOT AND RETURN
         LXI     D,5805H
         XRA     A
         STA     8067H
         INX     H
         CALL    CNVRT
         CALL    PACK
G16      MOV     E,M
         INX     H
         MOV     D,M
         INX     H
         PUSH    H
         LXI     H,0FFFFH
         CALL    CMPDH
         JZ      G17
         XCHG
         CALL    RUN
         POP     H
         JMP     G16
G17      POP     H
         POP     H
         RET
PACK     LDA     5802H
         MOV     D,A
         LDA     5803H
         MOV     E,A
         XCHG
         RET
CNVRT    PUSH    H          ;THIS SUB CONVERTS ASCII TO PACKED HEX
                                                         CHARECTORS
G21      MOV     A,M        ;GET A BYTE
         ANI     7FH        ;REMOVE MSB
         SUI     30H        ;SUBTRACT 30
         CPI     0AH        ;IF STILL GREATER THAN A THEN SUBTRACT
                                                             7 MORE
         JC      G22
         SUI     07H
G22      RLC
         RLC
         RLC
         RLC                ;ROTATE UP TO TOP NIBBLE
         MOV     B,A        ;STORE IN -B-
         INX     H          ;GET NEXT BYTE
```

```
        MOV     A,M
        ANI     7FH
        SUI     30H
        CPI     0AH
        JC      G23
        SUI     07H
G23     ORA     B       ;COMBINE TWO NIBBLES
        XTHL
        MOV     M,A
        INX     H
        XTHL
        CALL    CMPDH   ;WAS THAT THE LAST ONE?
        INX     H
        JNZ     G21     ;JUMP IF IT WASNT
        POP     H
        RET
TRNSMT  PUSH    B       ;THIS SUB TRANSMITS TO GROUND
        MOV     B,A
        CALL    FRM1
        LDA     802FH   ;GET STATUS OF FRAME
        ORI     04H     ;SET TTY FLAG
        STA     802FH
        MOV     A,B
        STA     8037H   ;STORE DATA INTO TTY LOCATION
G26     IN      69H     ;WAIT FOR TRANSMITTER READY
        RAR
        JNC     G26
        MOV     A,B
        OUT     68H
        POP     B
        RET
RECEV   IN      5BH     ;HAS TIME-OUT OCCURED?
        ANI     20H
        JNZ     G27     ;JUMP IF IT HAS
        IN      69H     ;CHECK TO SEE IF RECEIVER HAS SOME DATA
        ANI     02H
        JZ      RECEV
        IN      69H
        ANI     30H
        JNZ     G38
        IN      68H
        ORA     A
        RET
G38     MVI     A,17H
        OUT     69H
G27     STC
        RET
FRM1    LDA     802EH   ;WAITS FOR ONE FULL FRAME TO GO BY
        CPI     TMBYT2
        JNZ     FRM1
G41     LDA     802EH
        CPI     01H
```

```
        JNZ     G41
        RET
FRM16   MVI     B,-16
G42     CALL    FRM1
        INR     B
        JNZ     G42
        RET
        END
```

IV. APPENDIX B - GROUND CONTROL PROGRAMS

TABLE OF CONTENTS

IV. APPENDIX B - GROUND CONTROL PROGRAMS (continued)

TABLE OF CONTENTS

IV. APPENDIX B - GROUND CONTROL PROGRAMS (continued)

TABLE OF CONTENTS

GCU SUBROUTINES

RST 1

This subroutine is under the RESTART1 control. It updates the hexadecimal displays. The hexadecimal characters in the hex-buffer are transferred into the 8279 display/keyboard encoder.

INT 6.5

When the GCU operates in the "MAIN" routine a depression of any key sends the unit into this subroutine. All analog channels are reset to ZERO, displays are cleared, LED indicators are turned off and the control is transferred to the executive program.

INT 7.5

The interrupt is activitated by the PCM byte counter. The subroutine processes the PCM data. An 8 bit data word is transferred to the monitor port. An appropriate strobe pulse is generated. Also, the data is stored into one of two buffers. While one of the buffers is being filled, the data previously stored in the other buffer is processed as required.

CMBACK

This subroutine sets selected registers and flags to their initial conditions It determines which function is to be executed next, finds the starting address of the appropriate routine and transfers control to that program.

FRAME

This subroutine establishes the PCM frame length and the necessary word synchronization for decommutation of the pulse train. Frame length is determined by counting the number of eight bit words in two consecutive frames. When agreement between the two counts exists the task is completed by assuring that the frame length does not exceed the PCM buffer. Frame synchronization will not be achieved when the frame length exceeds 3F words.

YORN

When a YES or NO reply is required from the keyboard operator, this routine is executed while waiting for the answer. NO sets the CY=1, YES sets CY=0.

MODE

This is an interactive program which guides the operator through the necessary steps to compose the MODE BYTE of an instruction set to be transmitted to the Flight Control Unit (FCU). It also defines the amu stepping increment (BYTE 16H). In the MODE BYTE it sets the NO DOWN, AMU or BIAS SWEEP, HI-PASS, ACCUMULATION and SWITCHING mode bits.

BIASP and BIASS

These routines convert the 5 primary or secondary bias values to the ASC11 code and store these codes in the appropriate buffer locations before transmission to the FCU.

READ

The routine is used to read a character from a terminal or from the GCU keyboard. Upon an entry a buzzer is activiated. The character code is converted into the binary code acceptable to the system.

NMREAD

Accepts only ESCAPE (ESC), BACK SPACE (BS), CARRAGE RETURN (CR) and NUMBER entries. Any other entry forces a new reading attempt.

MOVE

Transfers data from one block of RAM into another block of equal size.

CMPDH

Compares DE registers with HL. HL>DE, CY=1; HL=DE, CY=0, Z=1: HL>DE, CY=0, Z=0.

ALREAD

Reads the ALPHA-NUMERIC characters into the display. Also responds to the BS, ESC, and CR entries.

BELL

Activates the buzzer for a preset number of loops=X.

DIRECT

This program allows the operator to address directly the entire memory space of the GCU. Some or all of the functions associated with the programming or manipulation of the memory may be selected to operate in the DIRECT mode. CR after "DIRECT" entry commmits all functions. An equals sign ("=") followed by a function after the "DIRECT" entry places only that function in the direct mode. More than one function may be commited to the direct mode by separating the functions with the "equals" sign. Since the keyboard on the GCU does not contain an "=" key, selection of individual functions for the direct access mode is only possible when a terminal is used. Also, the "DIRECT" mode must be used in conjuction with the "BINARY" mode where all numeric entries are in the hexadecimal notation.

ERROR

Clears all displays, activates the buzzer and writes "ERROR" in the alph-numeric display.

ENDIT

Ends a program and displays "END"

MESSAG

Displays a message indicated by a pointer and a message length counter. Also clears the display.

FILL

Fills memory starting with the address XXXX ending at YYYY with data ZZ.

CLEARD

Clears HEX displays.

CLEAR

Clears all displays.

MANY

Reads keyboard or terminal entries and displays same.

LOOKUP

Determines if the entered function in the alph-numeric buffer is valid. This is accomplished by comparing the entered code with the codes residing in a look-up table. When a match is found, the address and other parameters associated with the execution of that function are stored in appropriate buffers for further processing. Otherwise an error message is displayed. Since the GCU keyboard enters only one character, while a terminal is allowed eight characters to define a function, the validation processes differ.

ADDRES

Combines the entered numbers residing in the HEX display into one word representing an address or a single byte of data. The differentiation between the two is based upon the status of the address flag which is set or reset by the program which calls the ADDRES subroutine. The subroutine also warns the operator when the MODE (DIRECT, INDIRECT....) has not been defined. The final result of this subroutine is a word in the binary code acceptable to the machine.

INDRCT

The GCU is placed into an indirect addressing mode. Only 2400H to 2BFFH RAM locations (0-7FF entries) are accessible in this mode. When using a terminal separate functions may be defined to operate in this mode by entering "=" sign after the "INDRCT" entry and then separating the functions by the equals symbol. Both the BINARY and the DECIMAL entry modes may be used.

BNRY

Address and data entries and displays are in the hexadecimal notation. When using a terminal only, selected functions may be defined to operate in this mode. The process is described under INDRT.

DCML

Address and data entries and displays are in the decimal notation. Similar to the BNRY process. Confusion may result when two data bytes are displayed in the same readout. Also, DCML must be used with the INDRCT mode only!

MOVEM

Moves data from one block of memory to another block. It guides the operator through the process.

COMPA.

This program compares a block of data with another block of data. When a mismatch is encountered the address and the data byte in the second block are displayed. In this routine the first block is considered to be the "Known".

COMPD.

Compares a block of data with a given constant. Mismatched data byte with its address are displayed.

COMPSB.

Is used to display an address and associated data. Returns to the calling program/subroutine only after CR command is received.

ALTR.

This program is used to display and to alter data in a given memory location. The routine asks the operator for an address. It displays the address and the present data. New data may be entered in the GCU. It overwrites the old data upon CR. At the same time the address is incremented by one and displayed with the data present in the new location.

GETDAT.

Fetches one byte from an input device. When an error is detected the buzzer sounds and the routine is repeated unless the "return on CR" flag has been set by the calling program. In the latter case the routine is terminated and the control is returned to the calling program.

DECBIN.

Converts four digit decimal number into a 13 bit binary number.

DISPL.

This routine is only used in conjunction with a CRT terminal. It displays a block of memory starting with an XXX0 address followed by 8 bytes of data. The next 8 bytes will be preceeded by XXX8 address. The process is repeated until the end address (YYYF) is detected.

FEPROM.

Moves data from two predetermined memory blocks to a circuit simulating two EPROM'S. Two memory mapped output ports are utilized.

TTYLNK.

Maintains communications between the Flight Control and the Ground Control units. The routine waits for an ENQ character from the balloon package. Once that inquiry has been received, the routine repeats a message to the balloon unit three times. Each message starts with STX and ends with ETX characters. ASCII characters are used. An echo is expected from the airborne unit. A time limit is set on each communications attempt to avoid a permanent or a prolonged loop in case of a link failure.

BINDEC.

Converts four digit hexadecimal number into a four digit decimal number.

GOTO.

Transfers processor control to a new location.

DSDATA.

Displays a byte as two hexadecimal characters on the alphanumeric display When in decimal mode binary to decimal conversion is carried out and a three digit number is displayed

DSADDR.

Displays the address as a four digit number in the alphanumeric display or on CRT terminal. Adjustments for the direct or the decimal modes are performed.

BAUD.

Sets the baud reate to accommodate the terminal. Baud rates of 75, 110, 300, 600, 1200, 2400, and 9800 are possible.

NPAGE

Prepares the TTY Buffer to transmit the "Run a New Instruction Set" Command.

DUMP

Prepares the TTY Buffer to transmit the "Dump the RAM" Command.

CONT

Stores the "Continue" command into the TTY Buffer.

WAIT

Stores the "Wait" command into the TTY Buffer.

RATIO

Prompts the operator to supply the RATIO information to be transmitted to the FCU in conjuction with the NPAGE command.

MASK

Prompts the operator to supply the Mask byte information for transmission to the FCU with the NPAGE command.

IDNUM

Asks the operator for the program ID to be transmitted to the FCU as a part of the NPAGE command.

LOP

Asks for the number of loops that an instruction set has to be repeated. Part of the NPAGE command sequence.

TIME

Requests the "Dwell Time" information. NPAGE command sequence.

AMU

Information to define the AMU scan is requested. Part of the NPAGE command sequence.

INITIAL

Initializes the PCM decommutation system. Assigns the displays to the specified PCM frame words. Also, defines the qualifiers for the display updating process.

SWITCH

Switches control between the GCU and a terminal.

ASCONV.

Converts ASCII into system binary.

BINCON.

Converts system binary into ASCII.

FILLM.

Fills a memory block with a given byte.

TERMIN.

Reads data from terminal. When the "RETURN" flag is set, returns to calling routine without converting the data into the system binary code.

TRMOUT.

Transmits a data byte to terminal.

TRNSMT.

Transmitts data from GCU to the FCU via USART.

RECEV.

Receives data from FCU. Checks for the communications time limit.

CNVRT.

Converts a word into four ASCII characters. Stores the characters starting with the address indicated by another routine.

CNVRT1.

Converts a byte into two ASCII characters. Stores the characters starting with an address indicated by another routine.

RPAGE.

Prepares the TTY Buffer to transmit a "Run an Instruction Set" Command.

RBOOK.

Prepares the TTY Buffer to transmit a "Run a Program" Command.

GETLOC

Receives a data byte from operator. When the byte represents a number less than 40H an offset is added to that byte before returning to the calling routine. When this condition is not met error message is displayed and the CPU control is returned to the calling routine.

GETALL

Prompts the operator to define various parameters for each display or analog channel. This information is stored in the decom buffer for later use.

ALLDEF

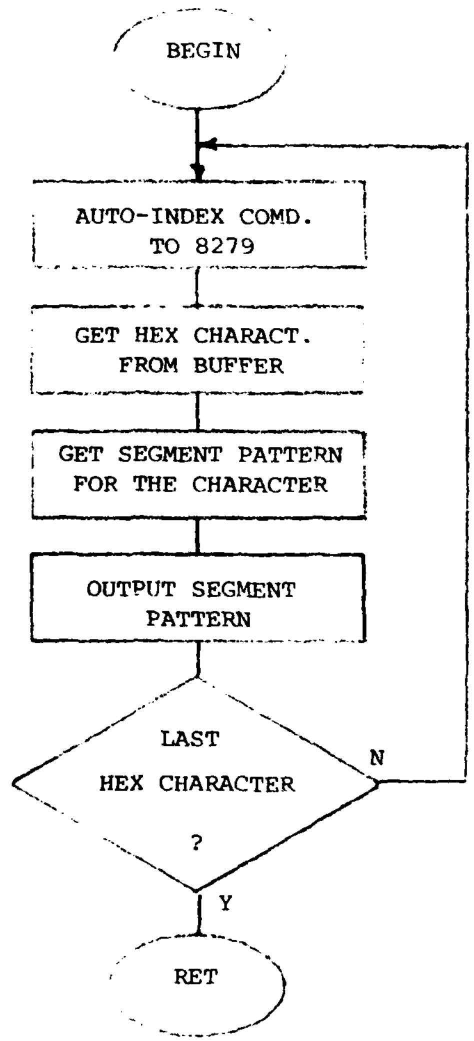
This routine checks the status of the various flags associated with the decommutation process. The SUBFRAME ID match to a predetermined number is also checked. When all of the necessary conditions are met for a particular byte/word of the PCM frame, the address of that byte/word located in a buffer designated during the initialization process is transmitted to the calling routine.

LDBUFF

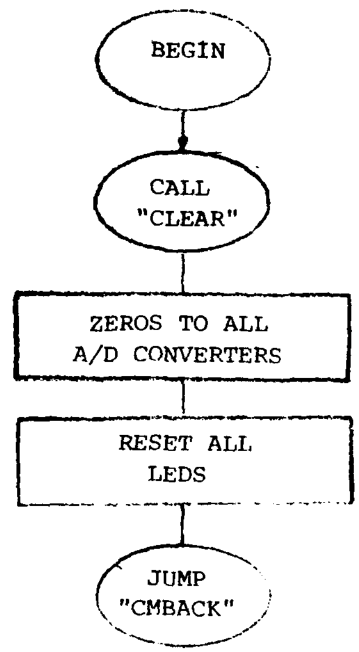
This routine waits for a PCM buffer to fill. Then the contents of that buffer are transferred to a new buffer for processing. The address of the Frame Sync Word is determined.

MAIN

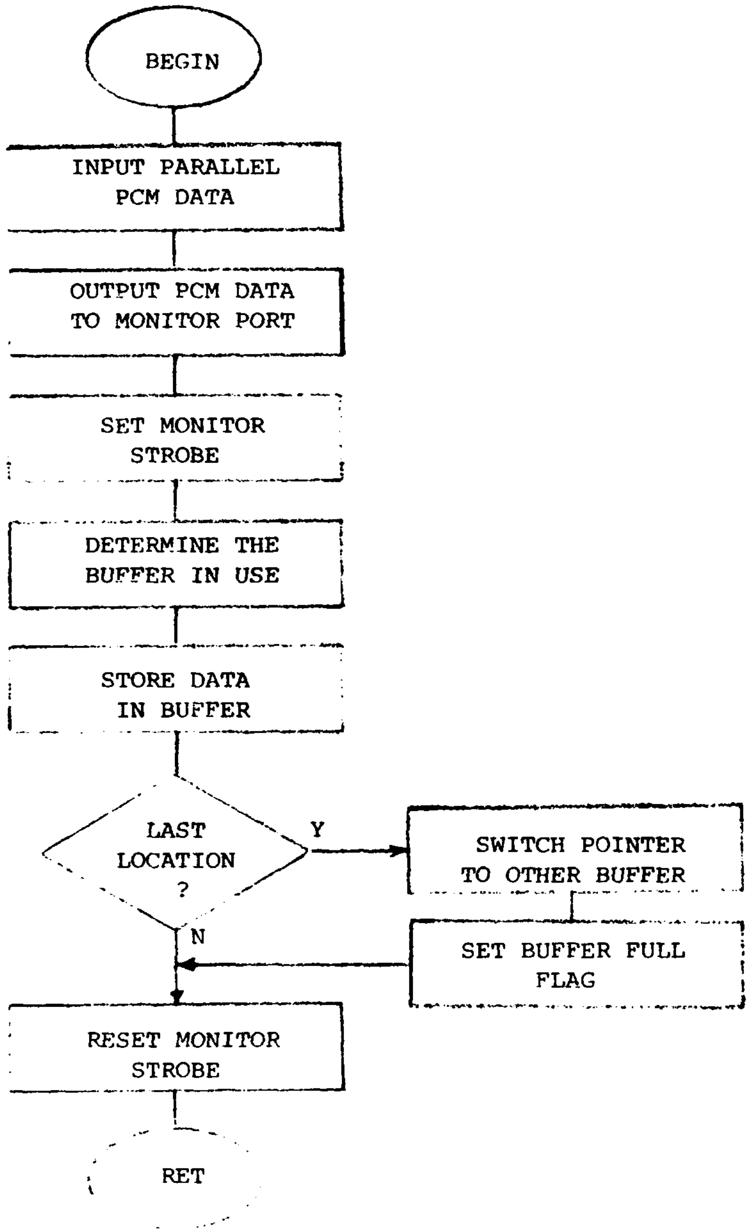
Forms the main PCM decommutating system. Transfers data to the designated displays. Receives and responds to messages. Determines and keeps track of frame and word synchronization.


BEGIN
AUTO-INDEX COMD. TO 8279
GET HEX CHARACT. FROM BUFFER
GET SEGMENT PATTERN FOR THE CHARACTER
OUTPUT SEGMENT PATTERN
LAST HEX CHARACTER ?
N
Y
RET

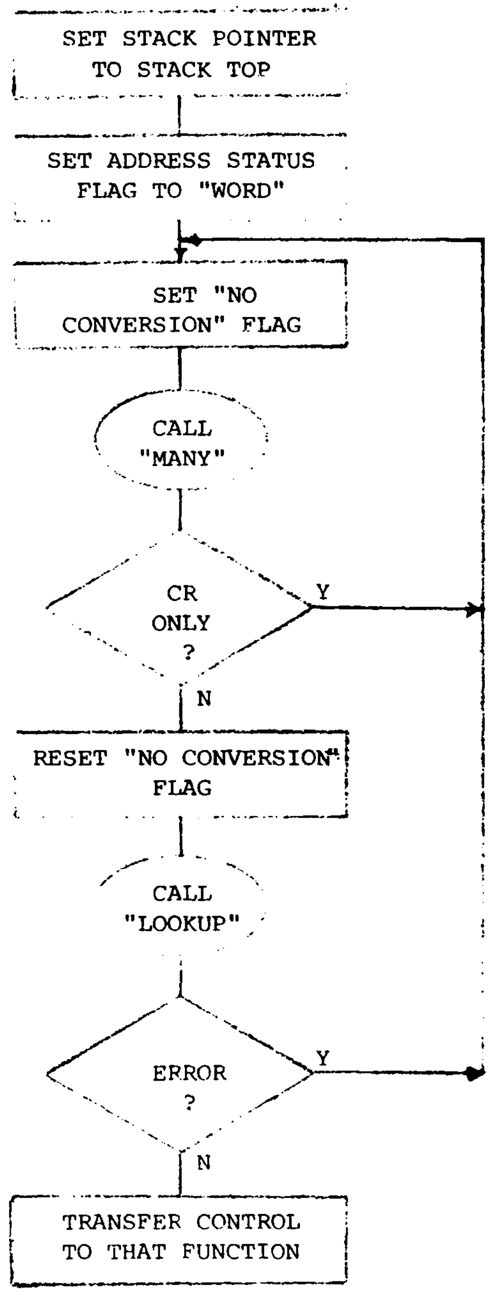
"RST1"


BEGIN
CALL "CLEAR"
ZEROS TO ALL A/D CONVERTERS
RESET ALL LEDS
JUMP "CMBACK"

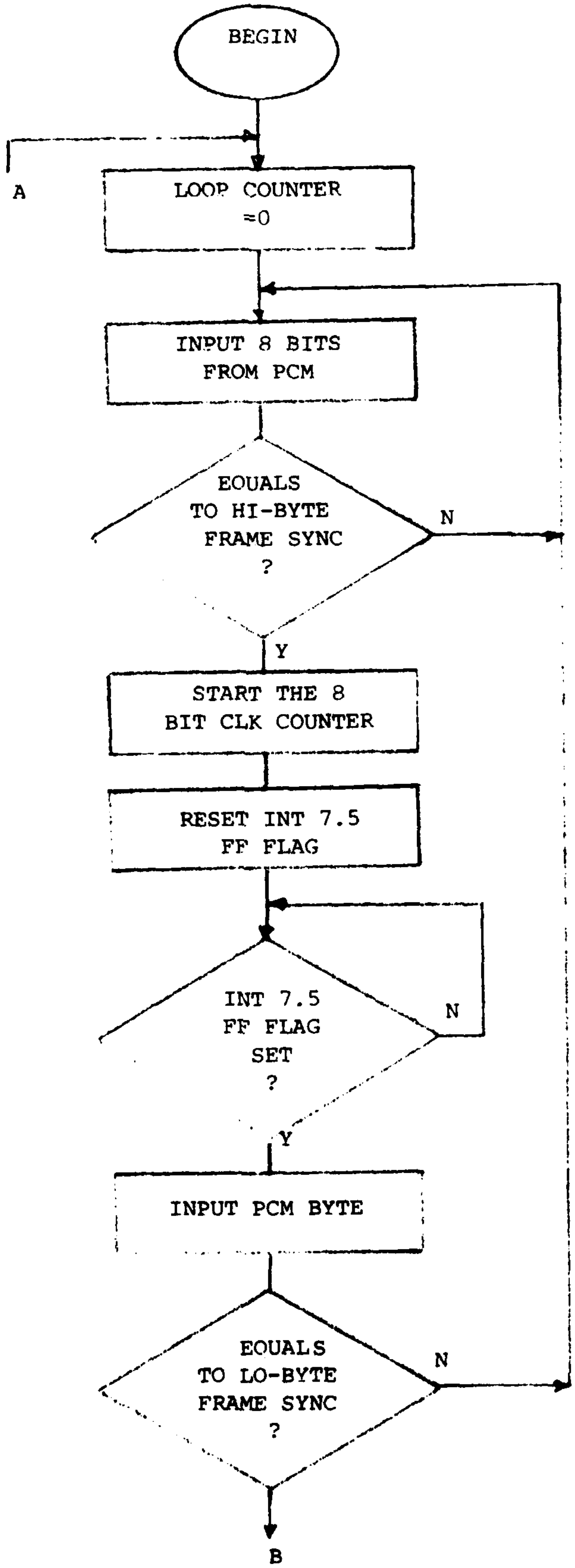
"INT6.5"



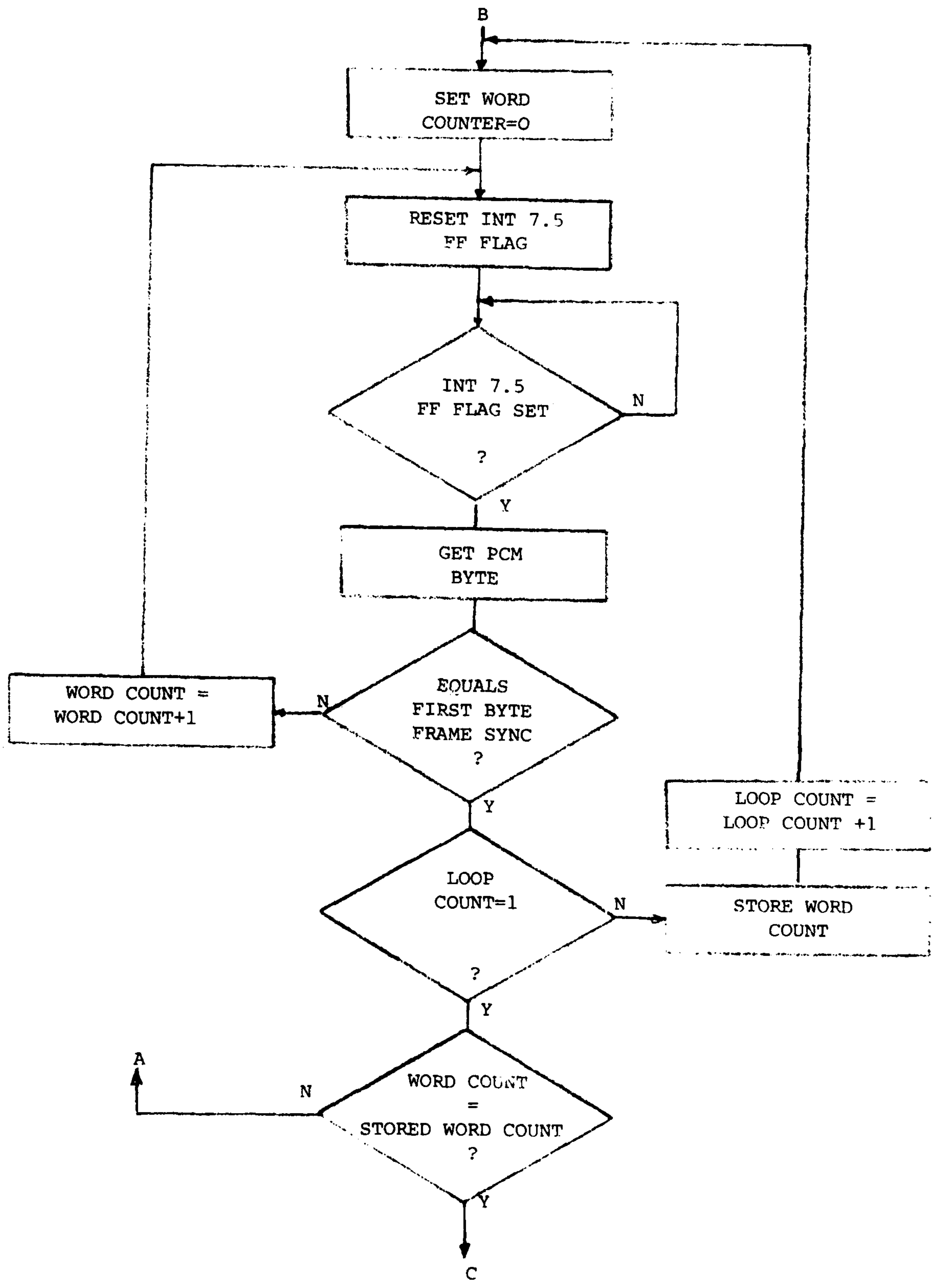
"INT7.5"



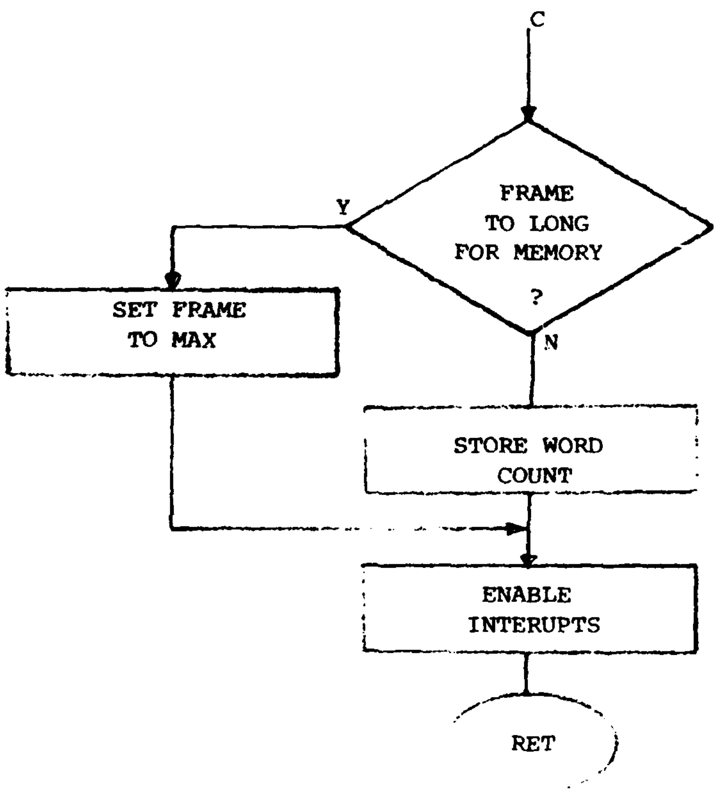
"CMBACK"


BEGIN
A
LOOP COUNTER =0
INPUT 8 BITS FROM PCM
EQUALS TO HI-BYTE FRAME SYNC ?
N
Y
START THE 8 BIT CLK COUNTER
RESET INT 7.5 FF FLAG
INT 7.5 FF FLAG SET ?
N
Y
INPUT PCM BYTE
EQUALS TO LO-BYTE FRAME SYNC ?
N
B

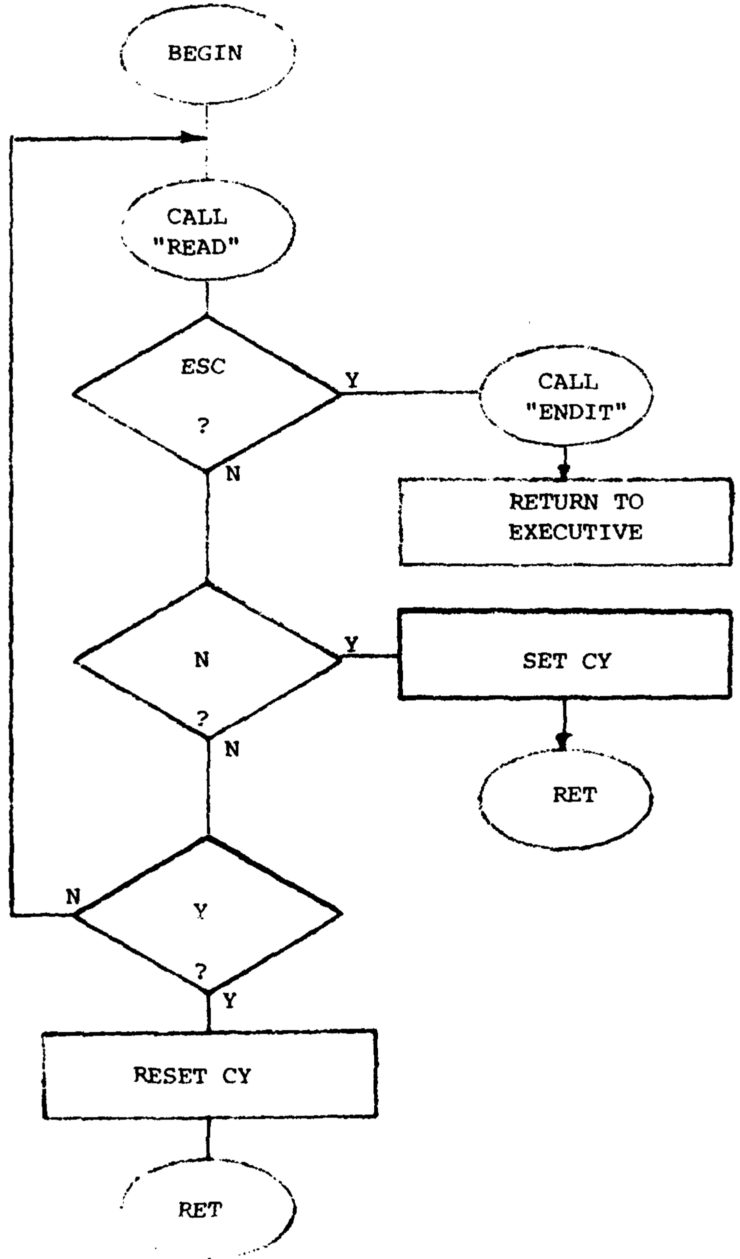
"FRAME"


B
SET WORD
COUNTER=0
RESET INT 7.5
FF FLAG
INT 7.5
FF FLAG SET
?
N
Y
GET PCM
BYTE
EQUALS
FIRST BYTE
FRAME SYNC
?
N
WORD COUNT =
WORD COUNT+1
Y
LOOP COUNT =
LOOP COUNT +1
LOOP
COUNT=1
?
N
STORE WORD
COUNT
Y
A
N
WORD COUNT
=
STORED WORD COUNT
?
Y
C

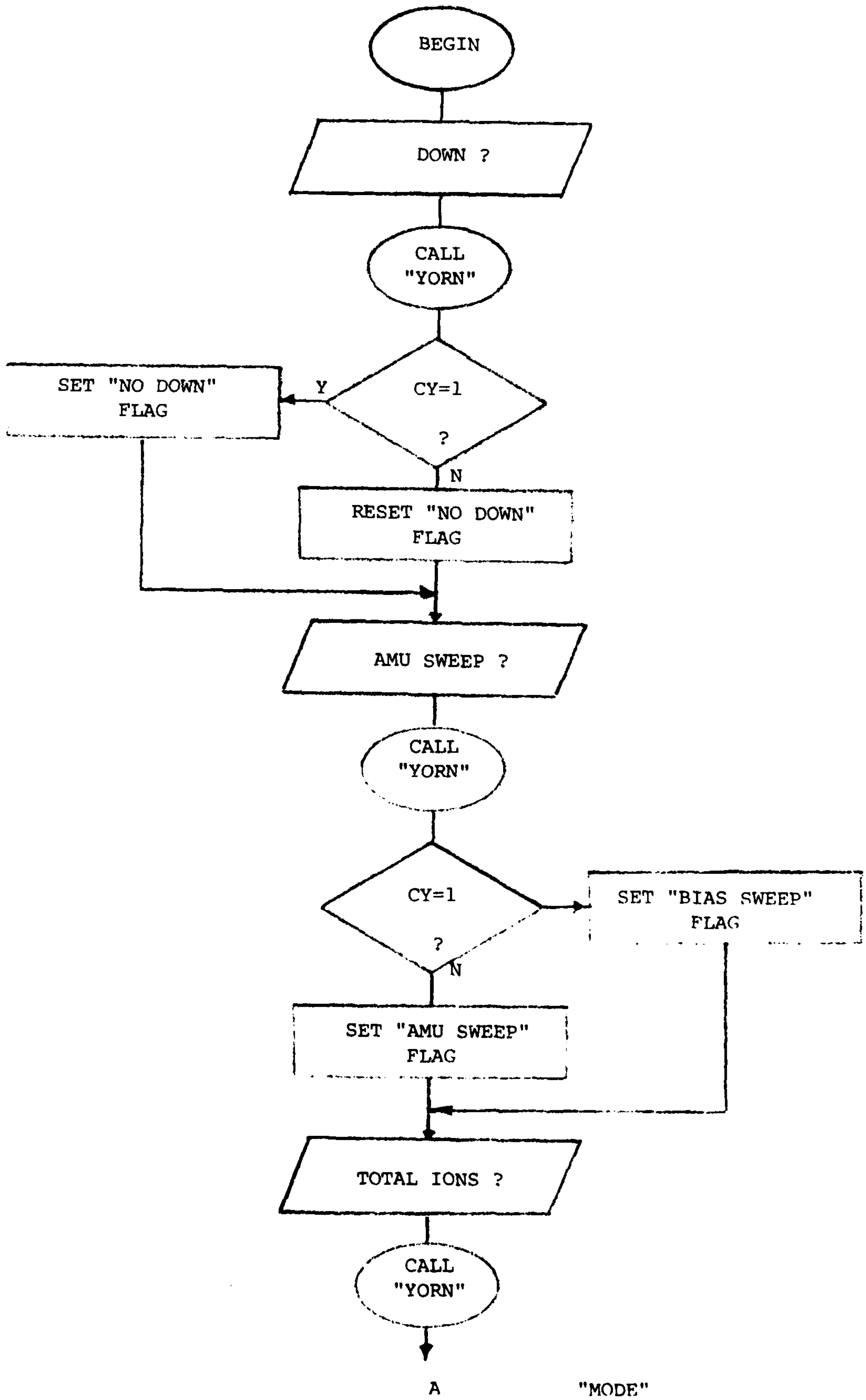
"FRAME" CONT.

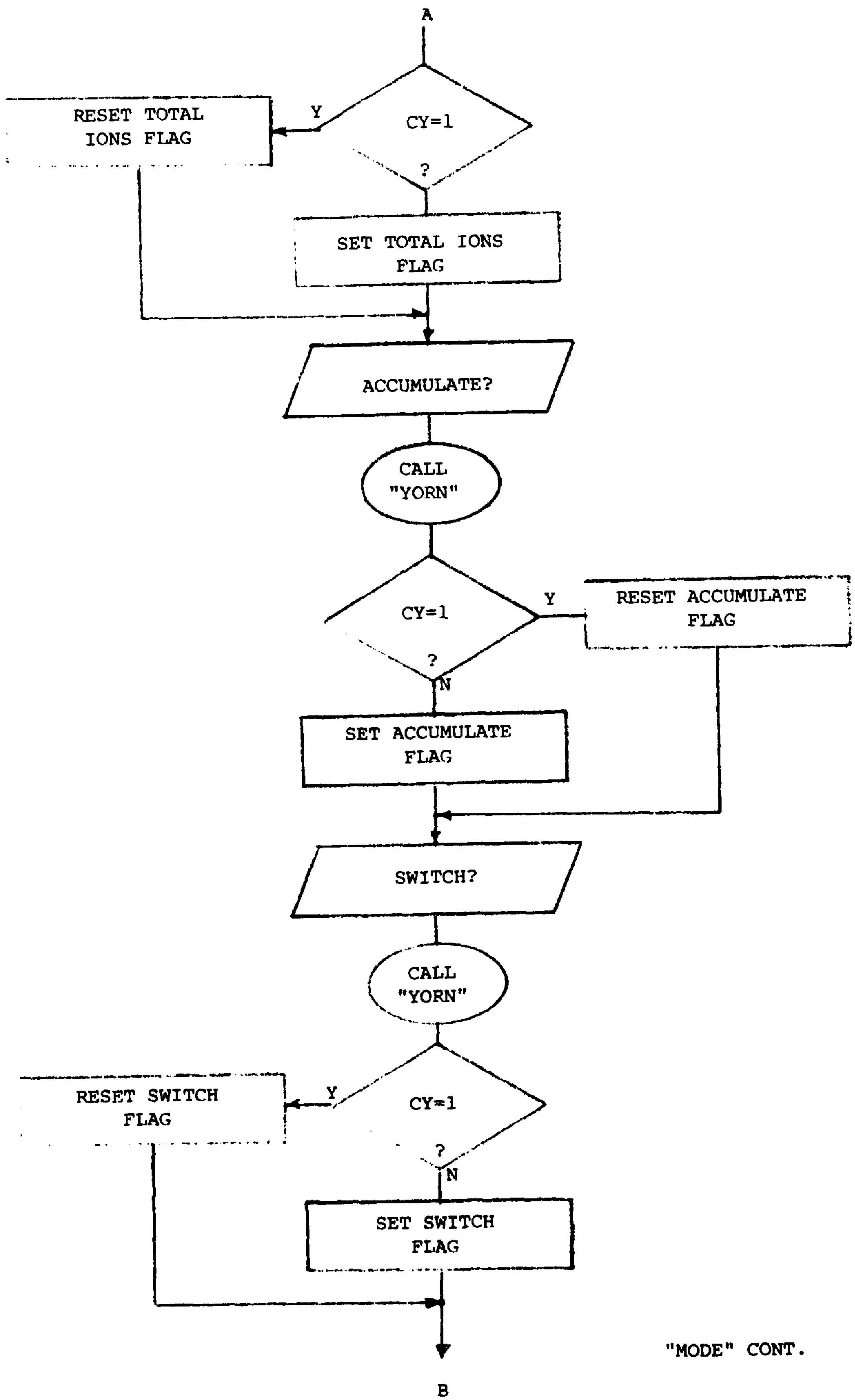


"FRAME" CONT.

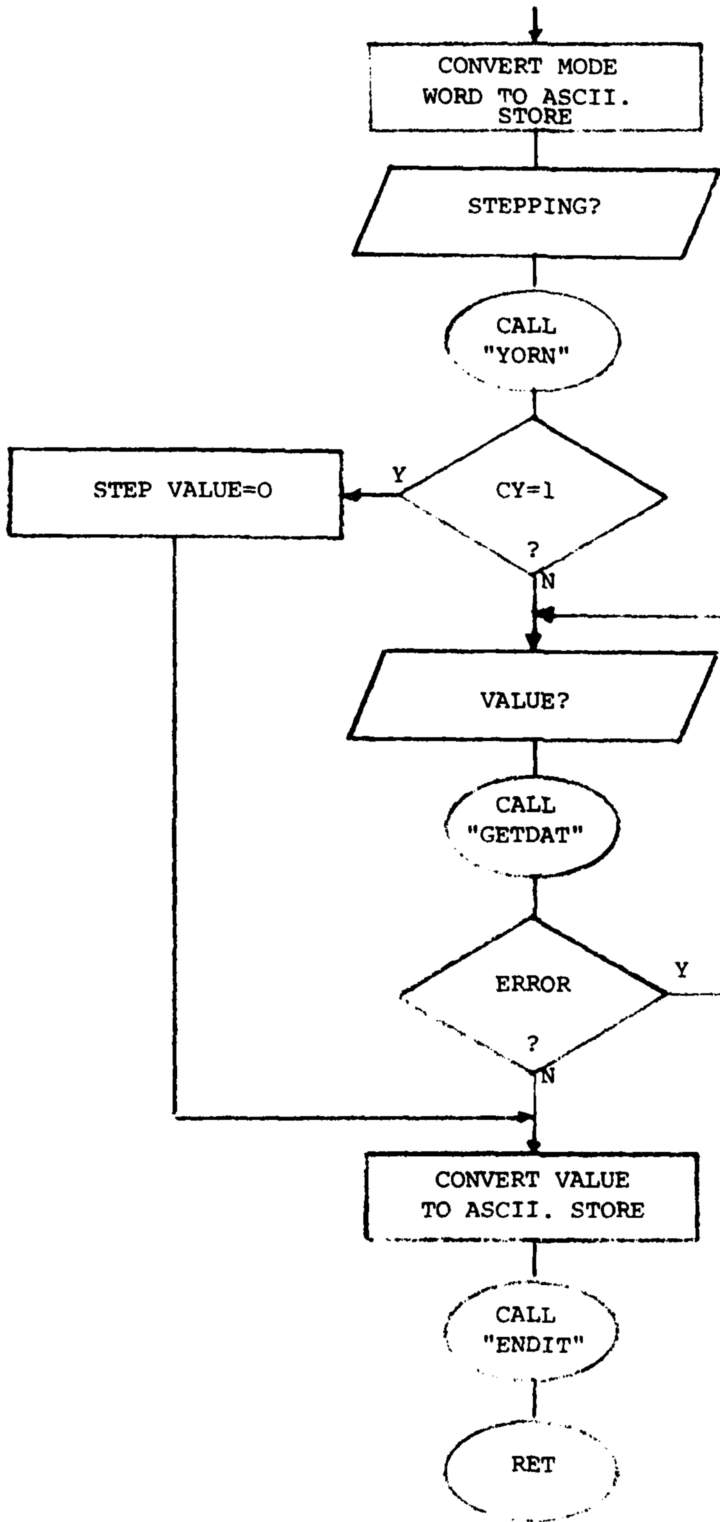


"YORN"


BEGIN
DOWN ?
CALL "YORN"
CY=1 ?
Y
N
SET "NO DOWN" FLAG
RESET "NO DOWN" FLAG
AMU SWEEP ?
CALL "YORN"
CY=1 ?
N
SET "BIAS SWEEP" FLAG
SET "AMU SWEEP" FLAG
TOTAL IONS ?
CALL "YORN"
A
"MODE"

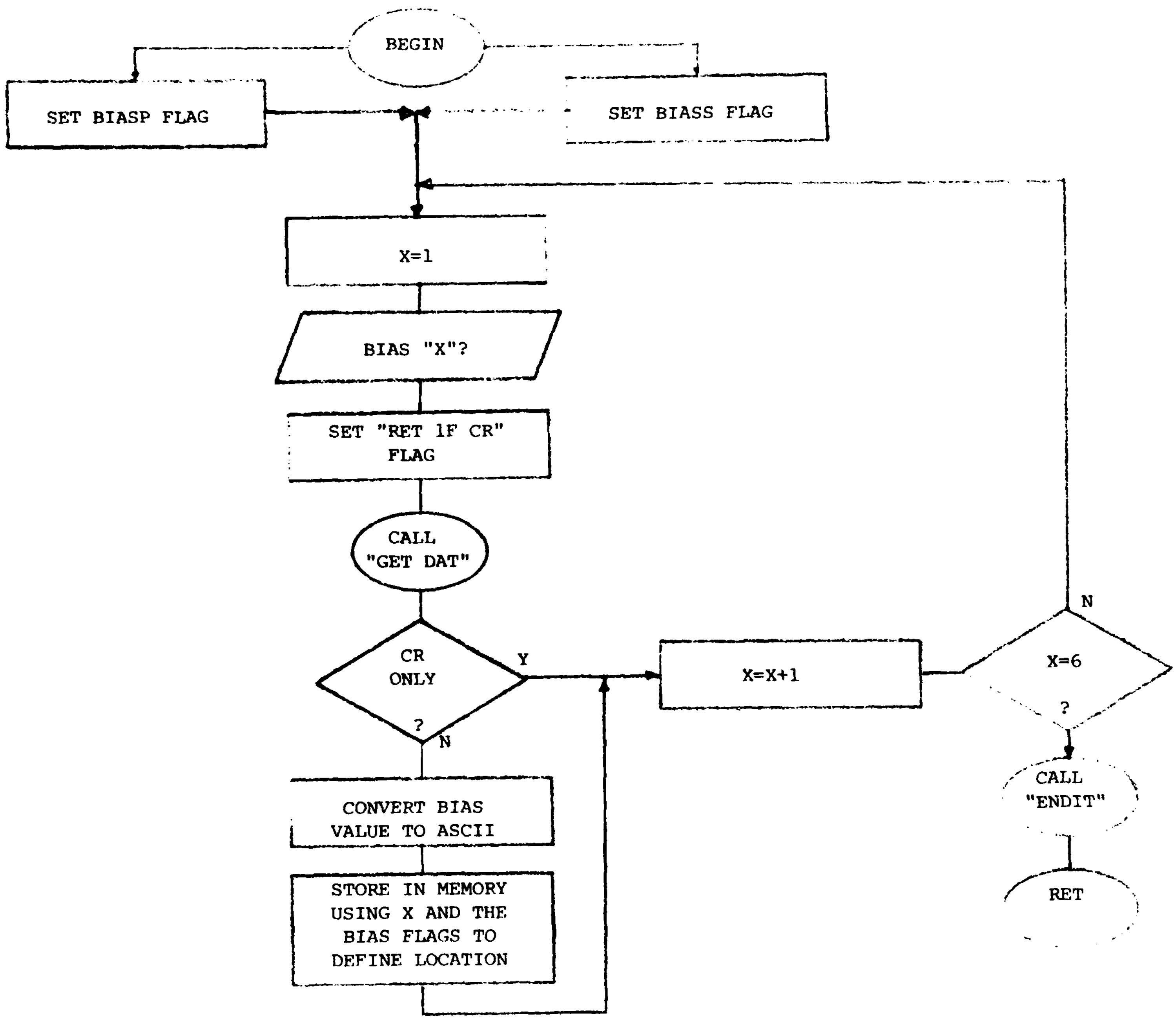

A
CY=1 ?
Y
RESET TOTAL IONS FLAG
SET TOTAL IONS FLAG
ACCUMULATE?
CALL "YORN"
CY=1 ?
Y
RESET ACCUMULATE FLAG
N
SET ACCUMULATE FLAG
SWITCH?
CALL "YORN"
CY=1 ?
Y
RESET SWITCH FLAG
N
SET SWITCH FLAG
B

"MODE" CONT.

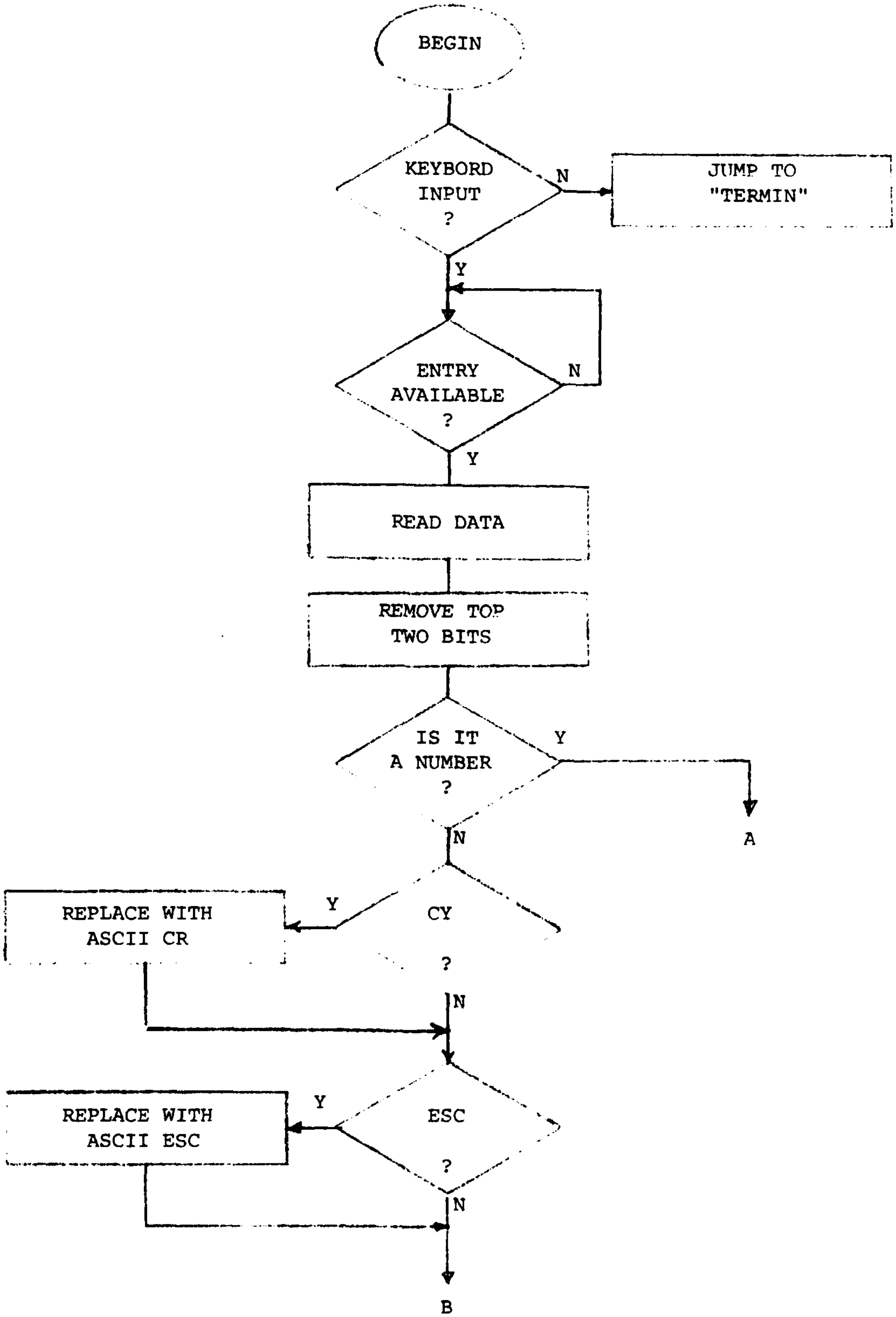


CONVERT MODE WORD TO ASCII. STORE
STEPPING?
CALL "YORN"
CY=1 ?
Y
N
STEP VALUE=O
VALUE?
CALL "GETDAT"
ERROR ?
Y
N
CONVERT VALUE TO ASCII. STORE
CALL "ENDIT"
RET

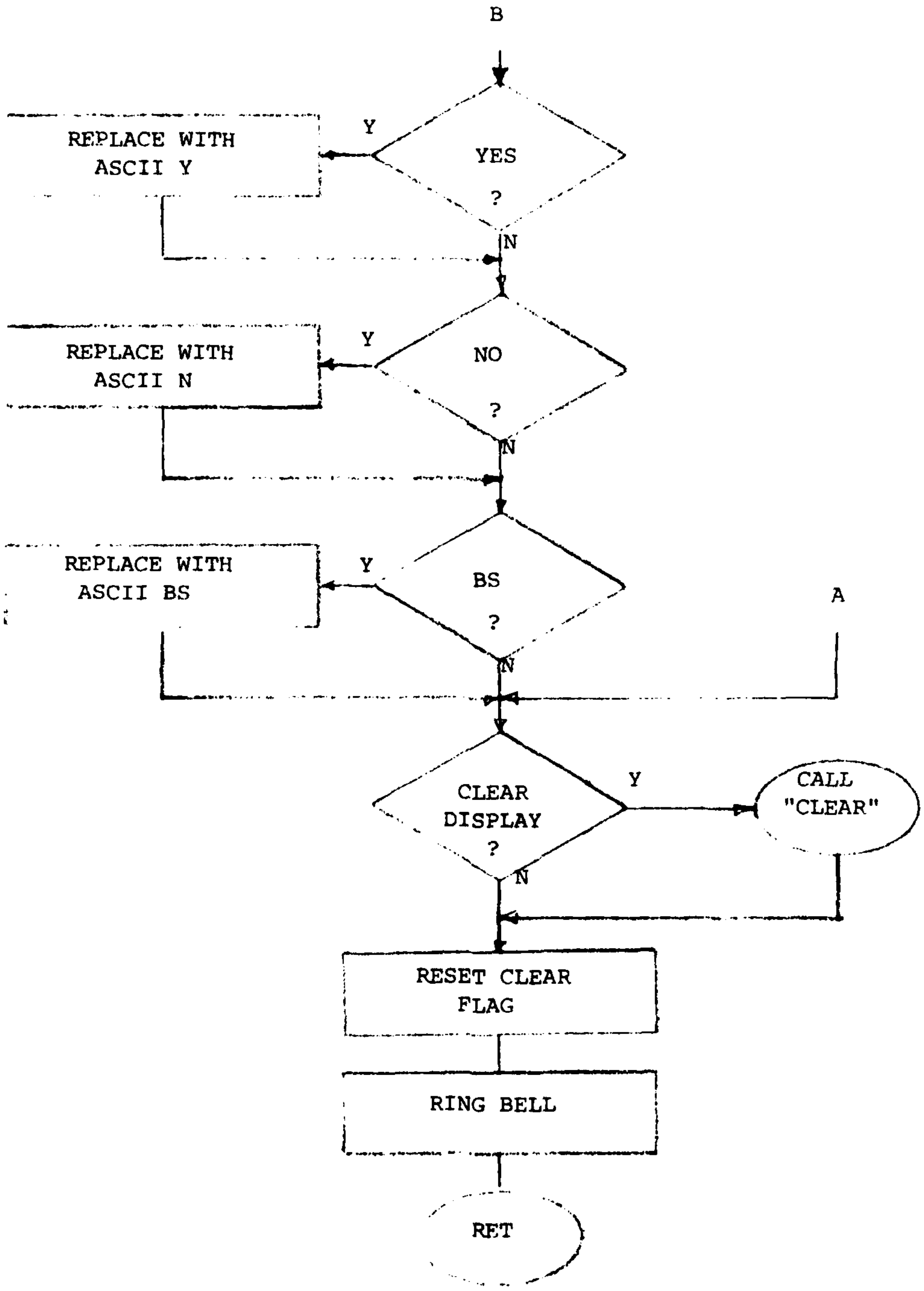
"MODE" CONT.



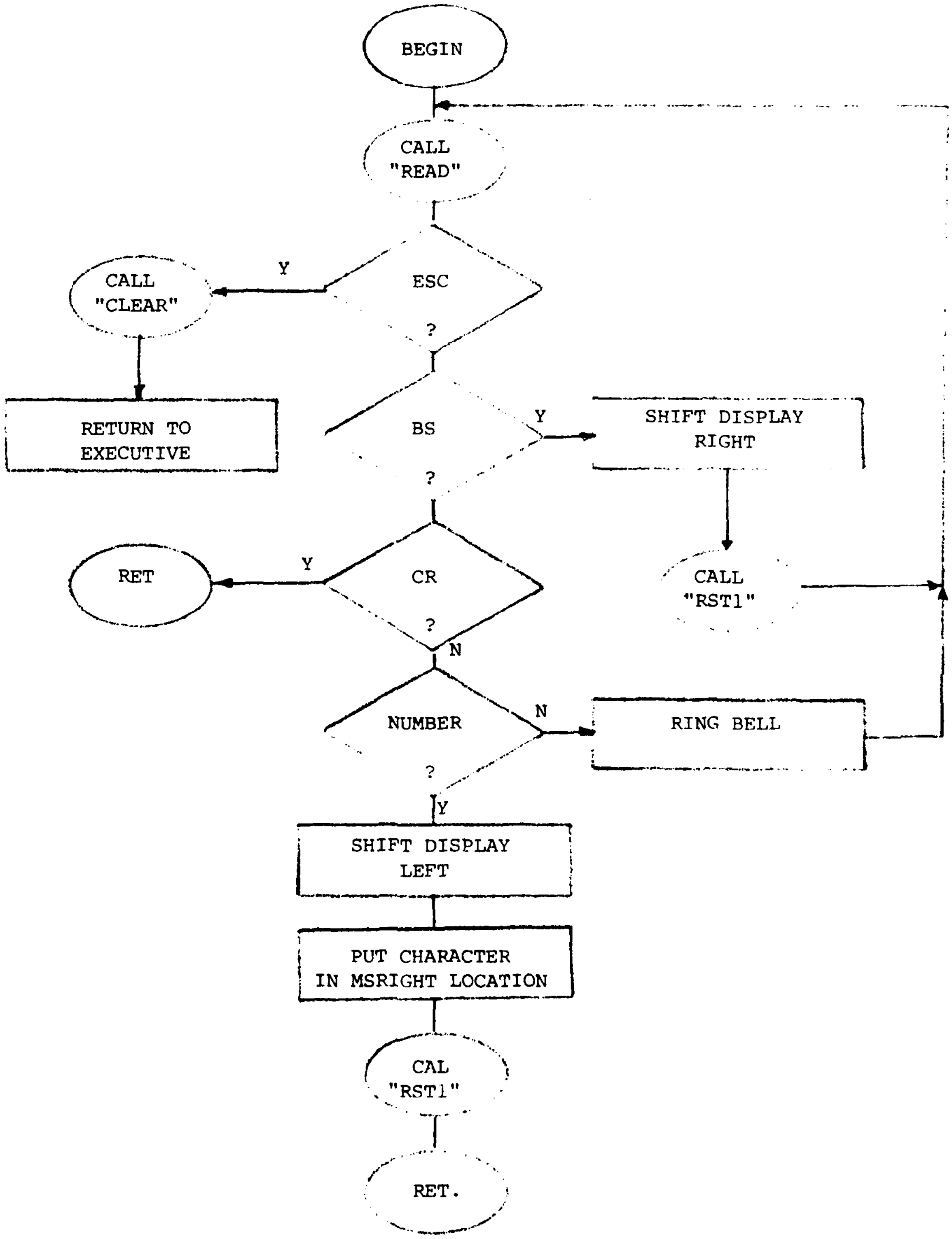
"BIASP" AND "BIASS"


BEGIN
KEYBORD INPUT ?
N
JUMP TO "TERMIN"
Y
ENTRY AVAILABLE ?
N
Y
READ DATA
REMOVE TOP TWO BITS
IS IT A NUMBER ?
Y
A
N
CY ?
Y
REPLACE WITH ASCII CR
N
ESC ?
Y
REPLACE WITH ASCII ESC
N
B

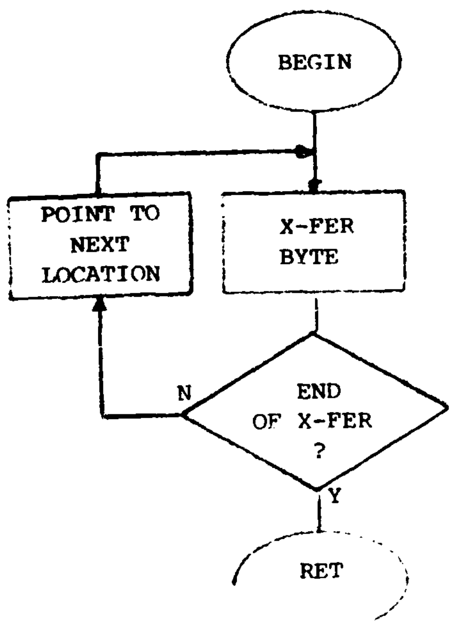
"READ"


B
REPLACE WITH ASCII Y
Y
YES ?
N
REPLACE WITH ASCII N
Y
NO ?
N
REPLACE WITH ASCII BS
Y
BS ?
N
A
CLEAR DISPLAY ?
Y
CALL "CLEAR"
N
RESET CLEAR FLAG
RING BELL
RET

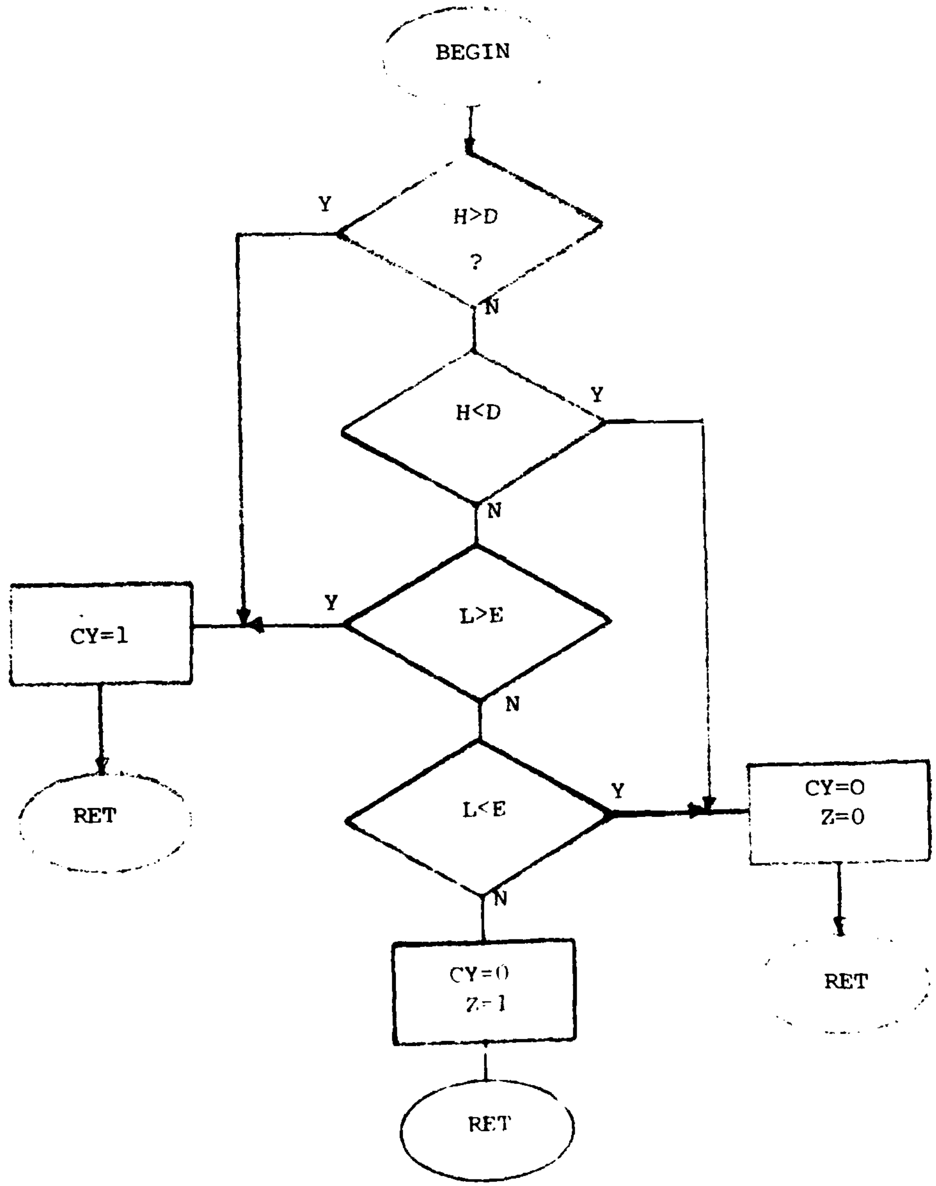
"READ" CONT.


BEGIN
CALL "READ"
ESC ?
Y
CALL "CLEAR"
RETURN TO EXECUTIVE
BS ?
Y
SHIFT DISPLAY RIGHT
CALL "RST1"
CR ?
Y
RET
N
NUMBER ?
N
RING BELL
Y
SHIFT DISPLAY LEFT
PUT CHARACTER IN MSRIGHT LOCATION
CAL "RST1"
RET.

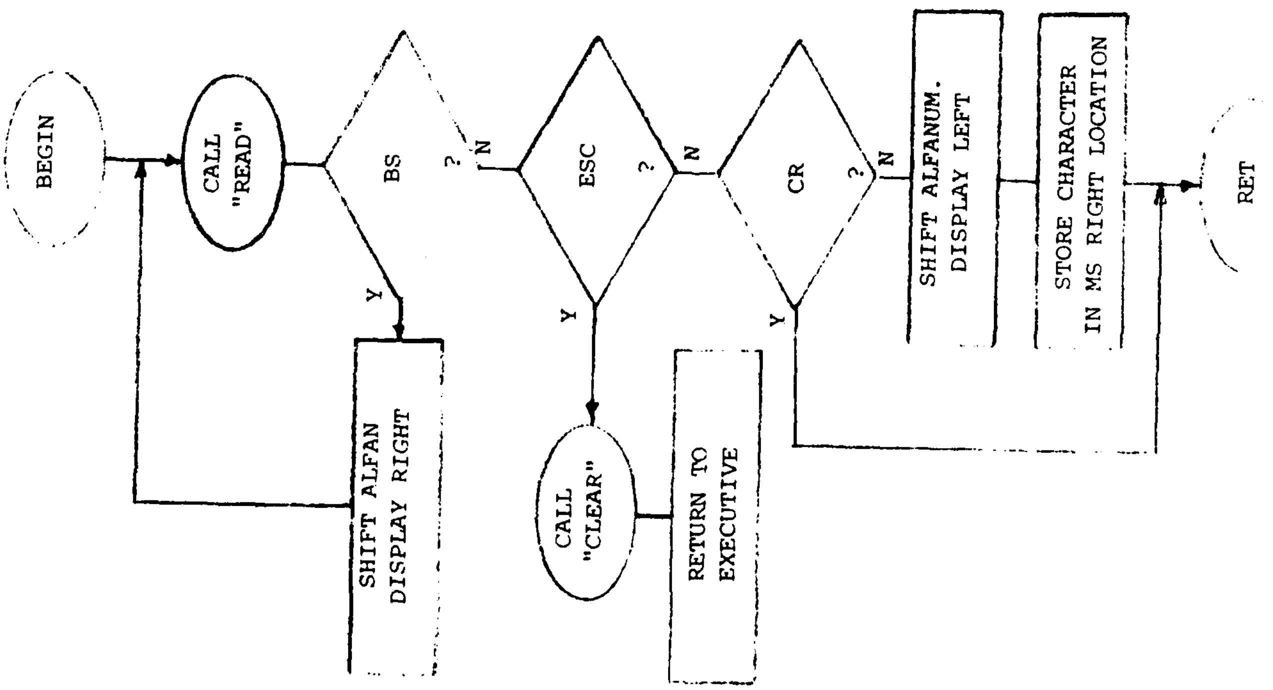
"NMREAD"


BEGIN
X-FER
BYTE
END
OF X-FER
?
Y
N
RET
POINT TO
NEXT
LOCATION

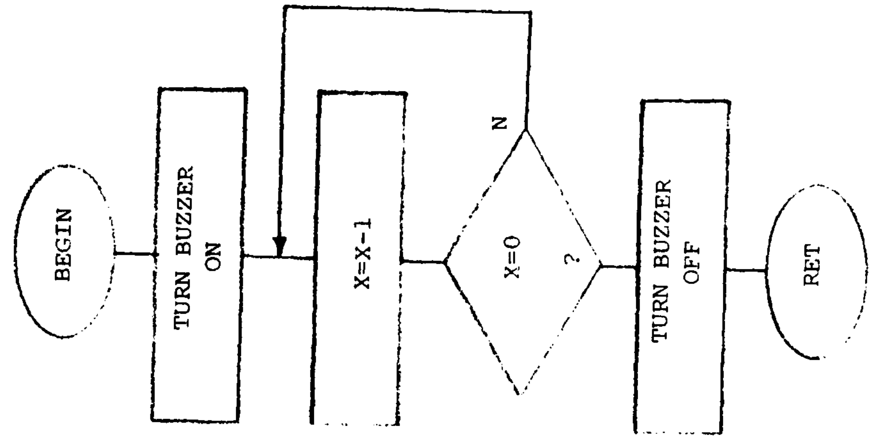
"MOVE"


BEGIN
H>D
?
Y
N
H<D
Y
N
L>E
Y
N
L<E
Y
N
CY=1
RET
CY=0
Z=0
RET
CY=0
Z=1
RET

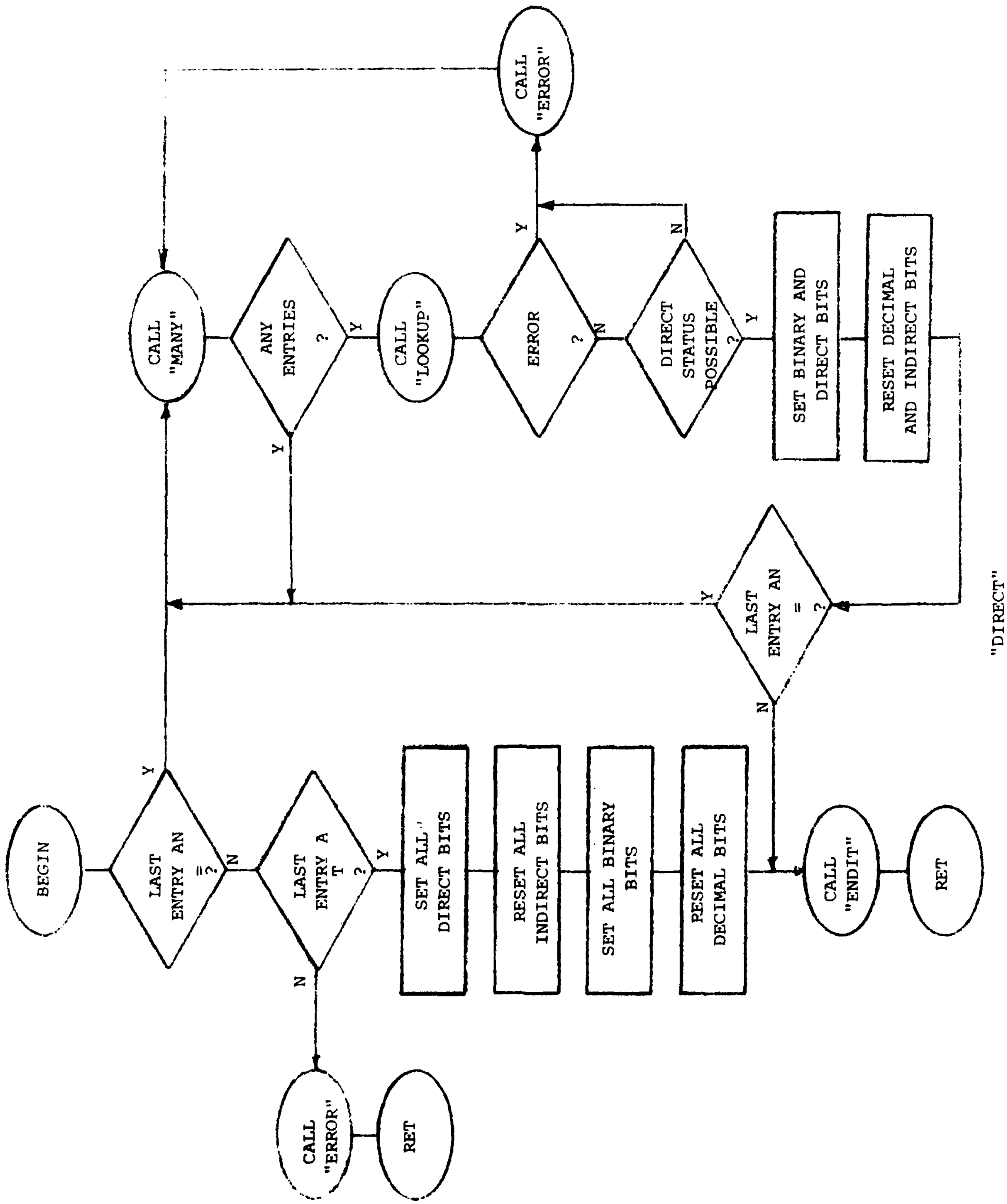
"CMPDH

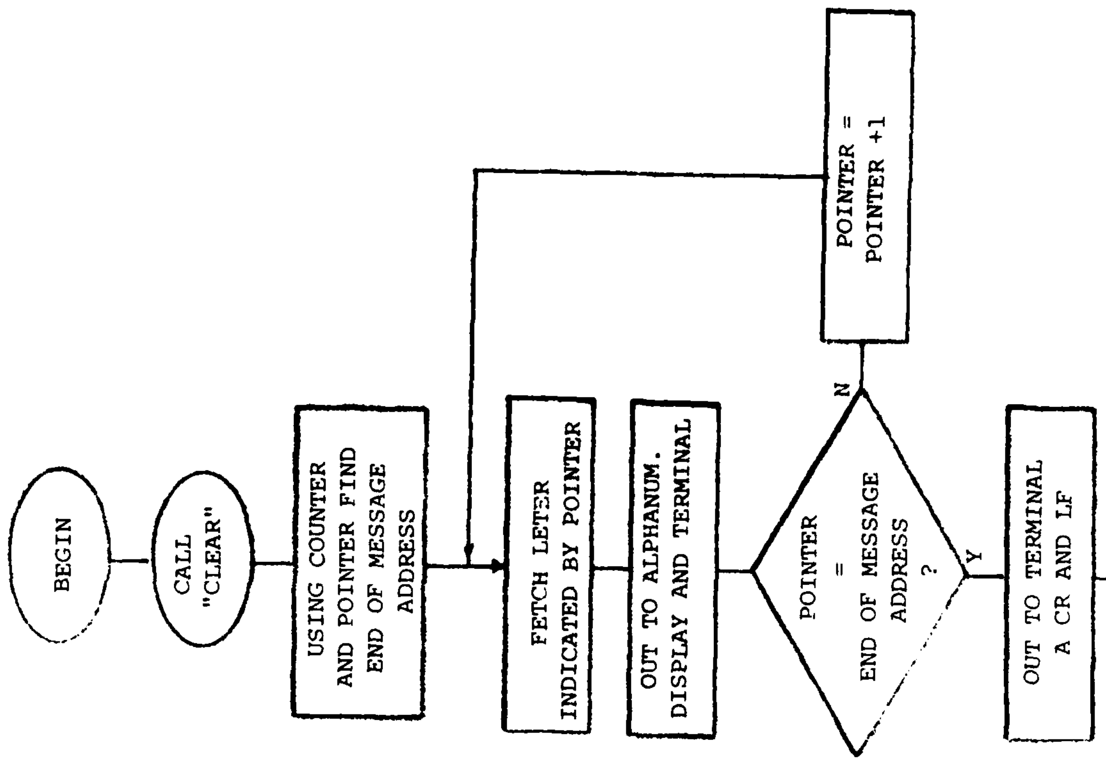

BEGIN
CALL "READ"
BS ?
Y
N
SHIFT ALFAN DISPLAY RIGHT
ESC ?
Y
N
CALL "CLEAR"
RETURN TO EXECUTIVE
CR ?
Y
N
SHIFT ALFANUM. DISPLAY LEFT
STORE CHARACTER IN MS RIGHT LOCATION
RET

"ALREAD"

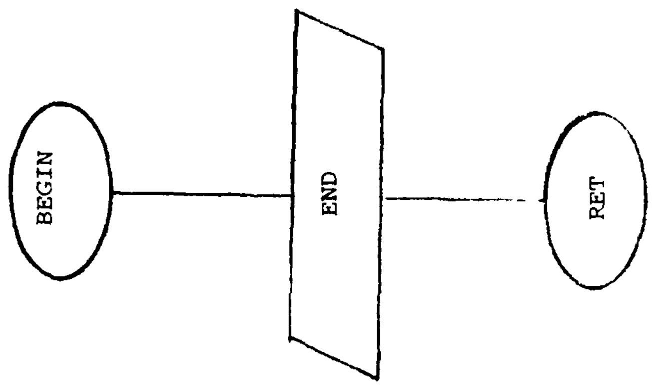

BEGIN
TURN BUZZER ON
X=X-1
X=0 ?
N
TURN BUZZER OFF
RET

"BELL"

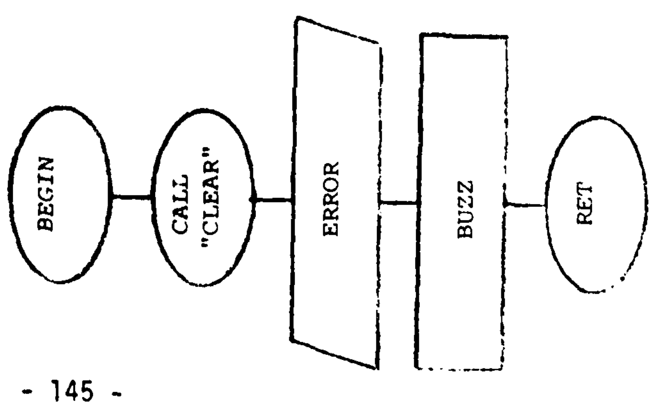

BEGIN
LAST ENTRY AN = ?
LAST ENTRY A T ?
CALL "ERROR"
RET
SET ALL DIRECT BITS
RESET ALL INDIRECT BITS
SET ALL BINARY BITS
RESET ALL DECIMAL BITS
CALL "ENDIT"
RET
CALL "MANY"
ANY ENTRIES ?
CALL "LOOKUP"
ERROR ?
CALL "ERROR"
DIRECT STATUS POSSIBLE ?
SET BINARY AND DIRECT BITS
RESET DECIMAL AND INDIRECT BITS
LAST ENTRY AN = ?
"DIRECT"
Y
N



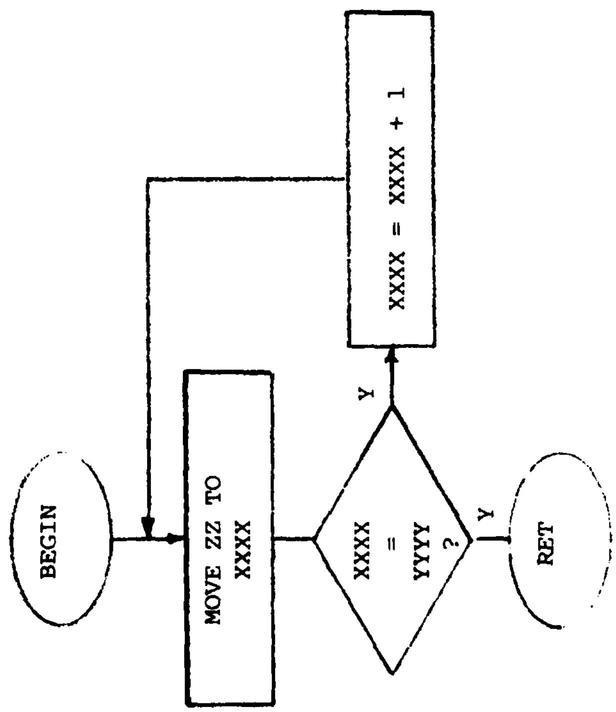
"MESSAG"



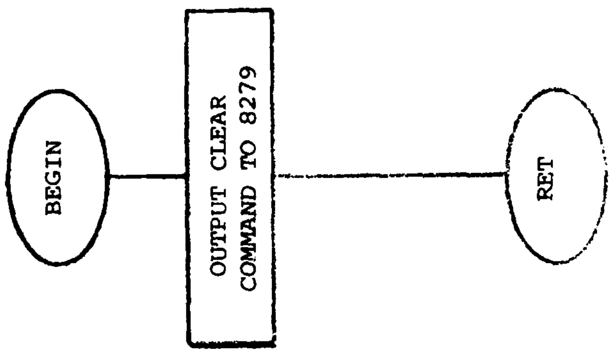
"ENDIT"



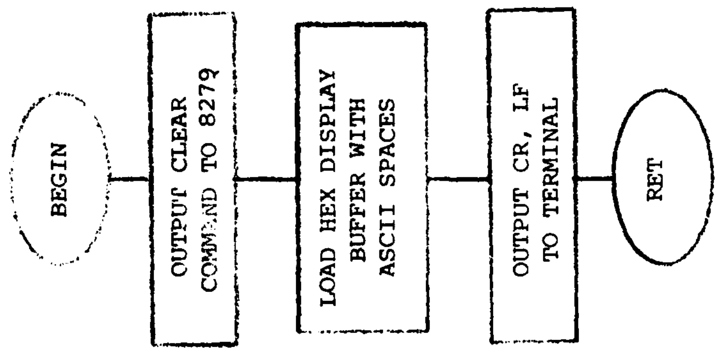
"ERROR"



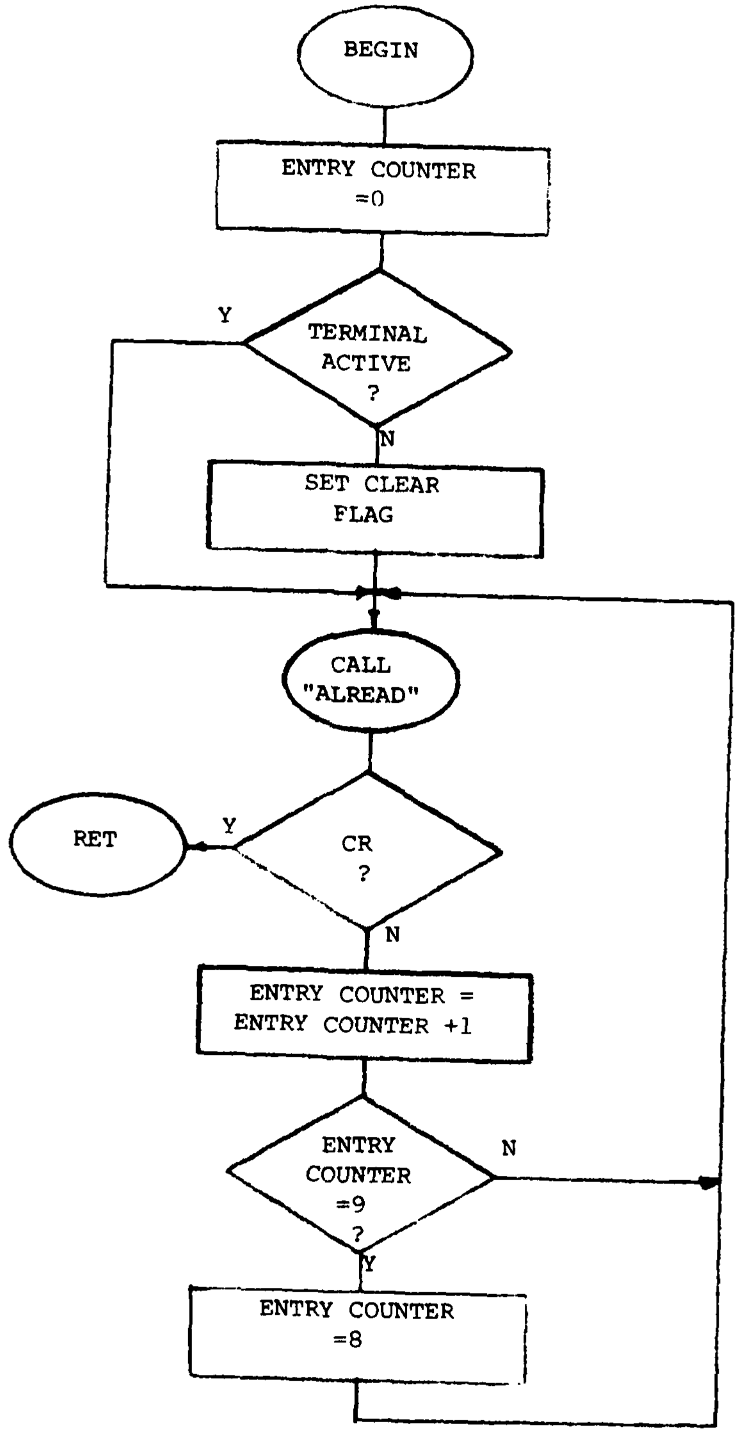
"FILL"



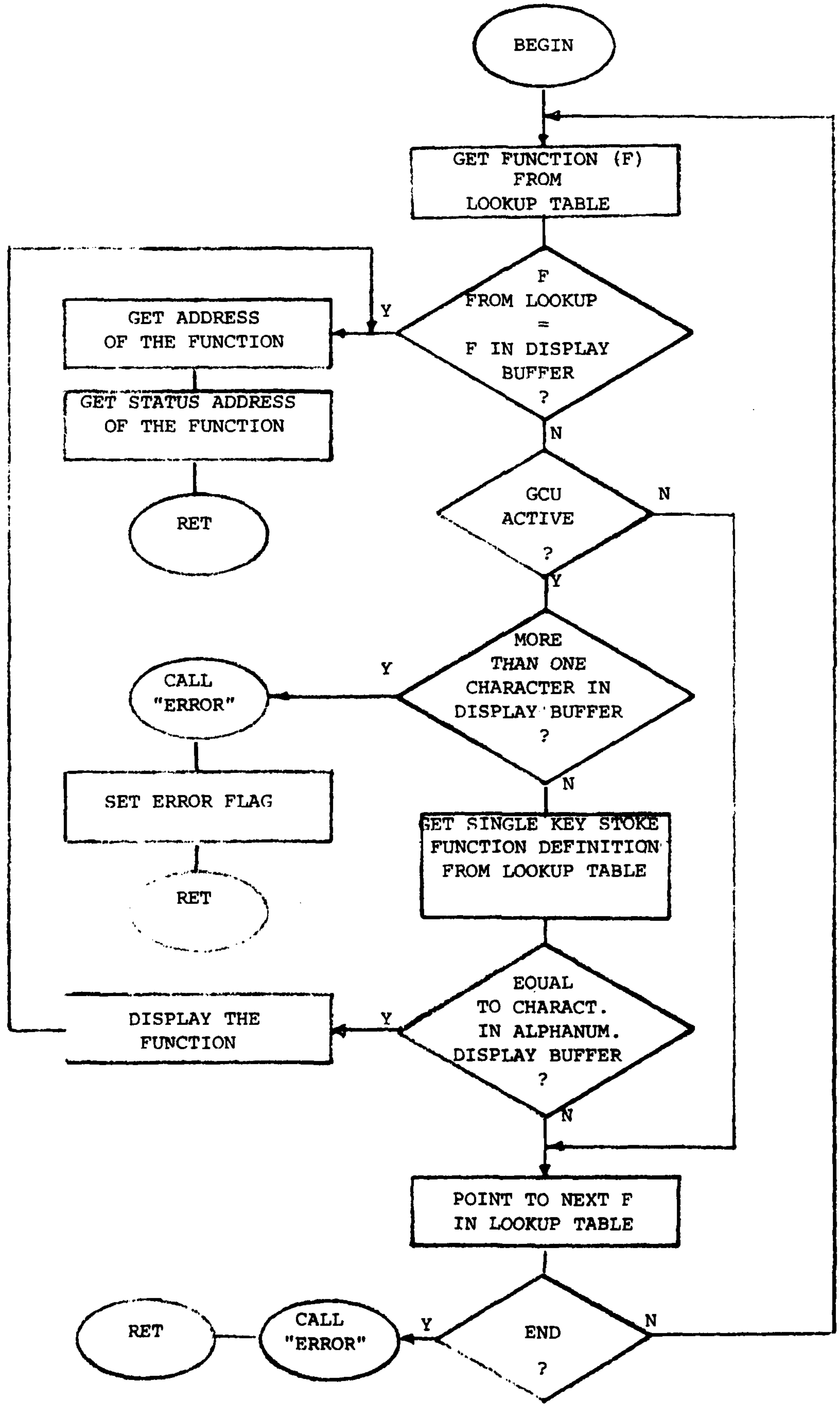
"CLEARB"



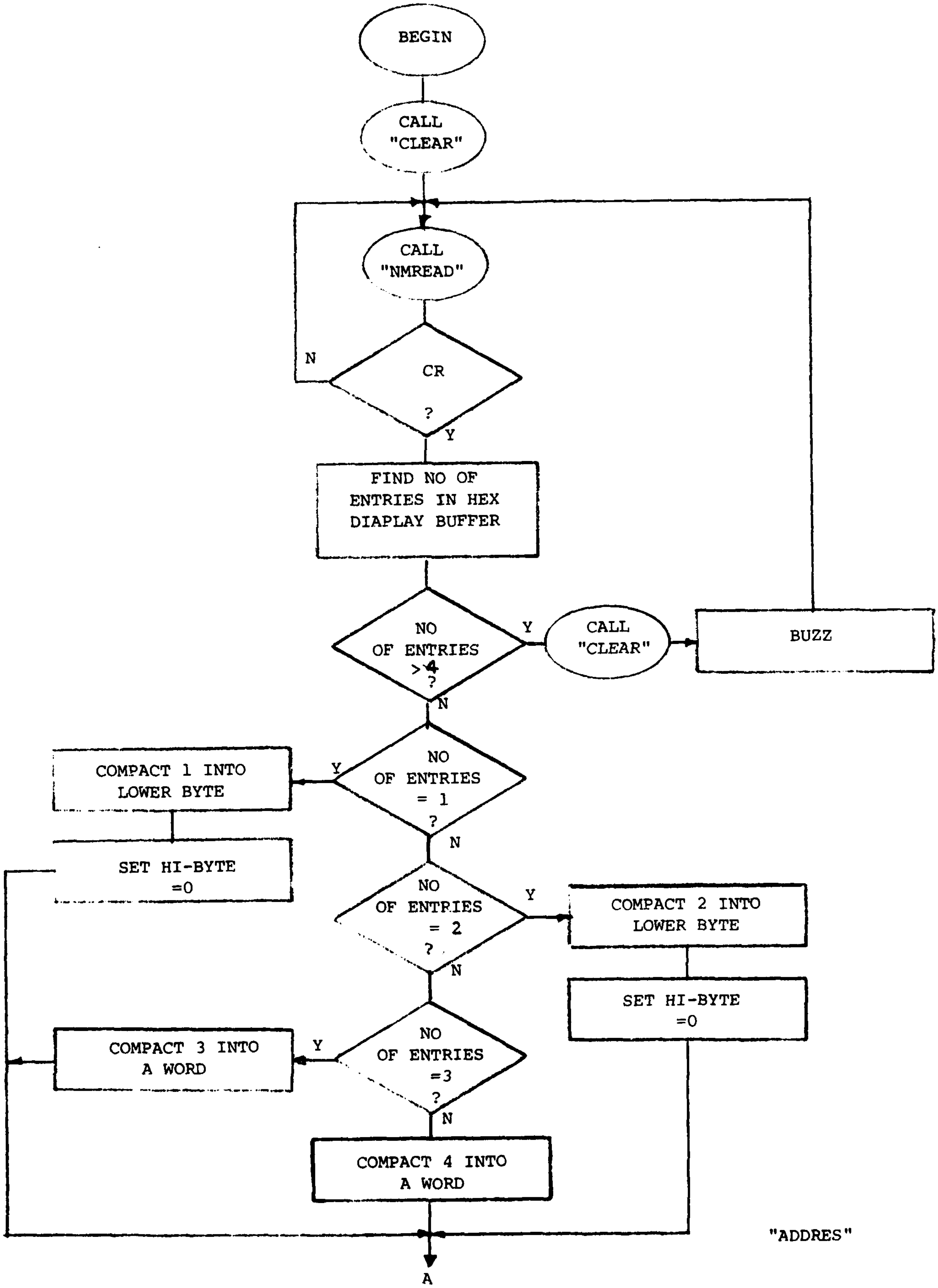
"CLEAR"

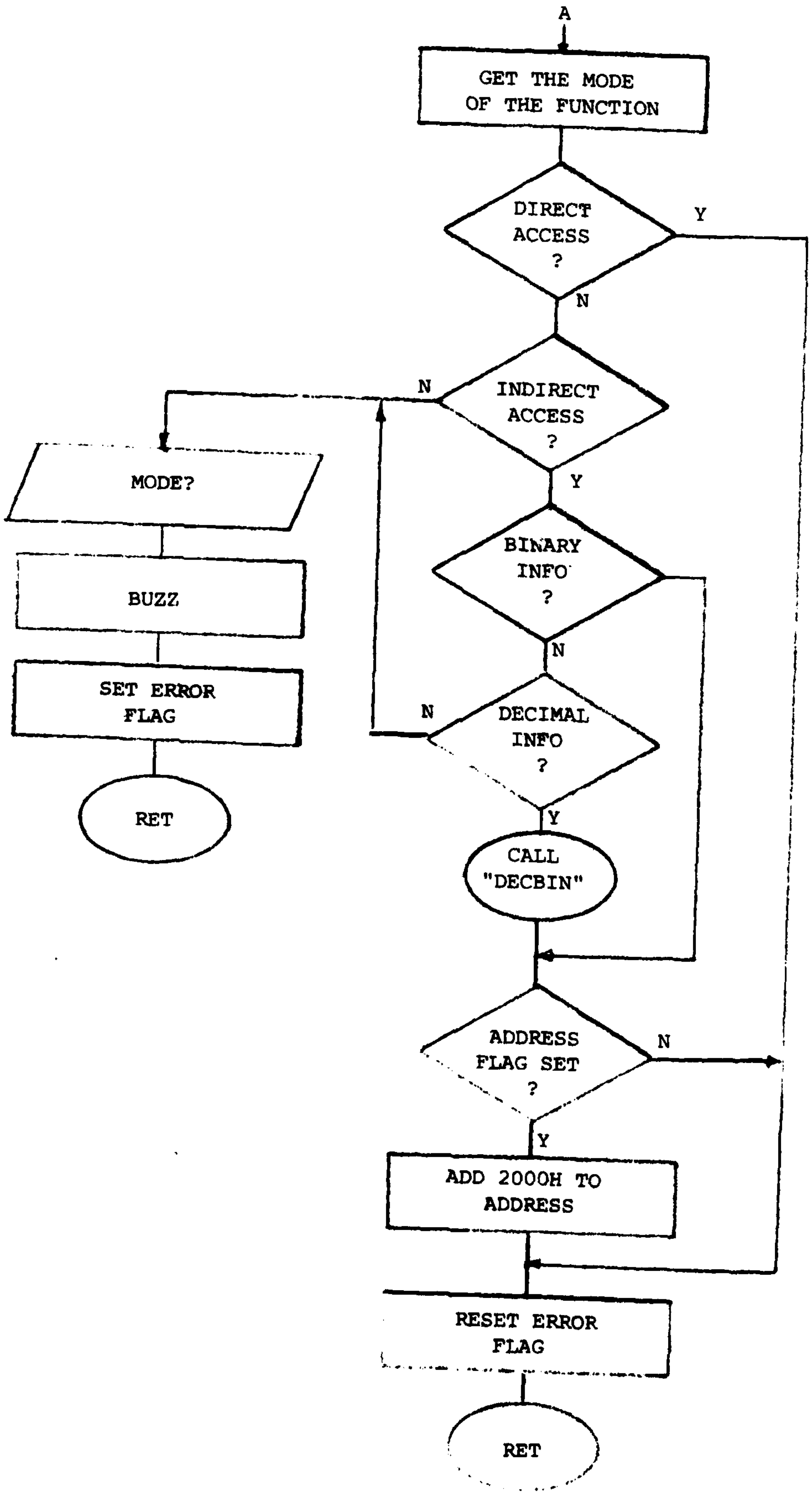

BEGIN
ENTRY COUNTER =0
TERMINAL ACTIVE ?
Y
N
SET CLEAR FLAG
CALL "ALREAD"
RET
Y
CR ?
N
ENTRY COUNTER = ENTRY COUNTER +1
ENTRY COUNTER =9 ?
N
Y
ENTRY COUNTER =8

"MANY"

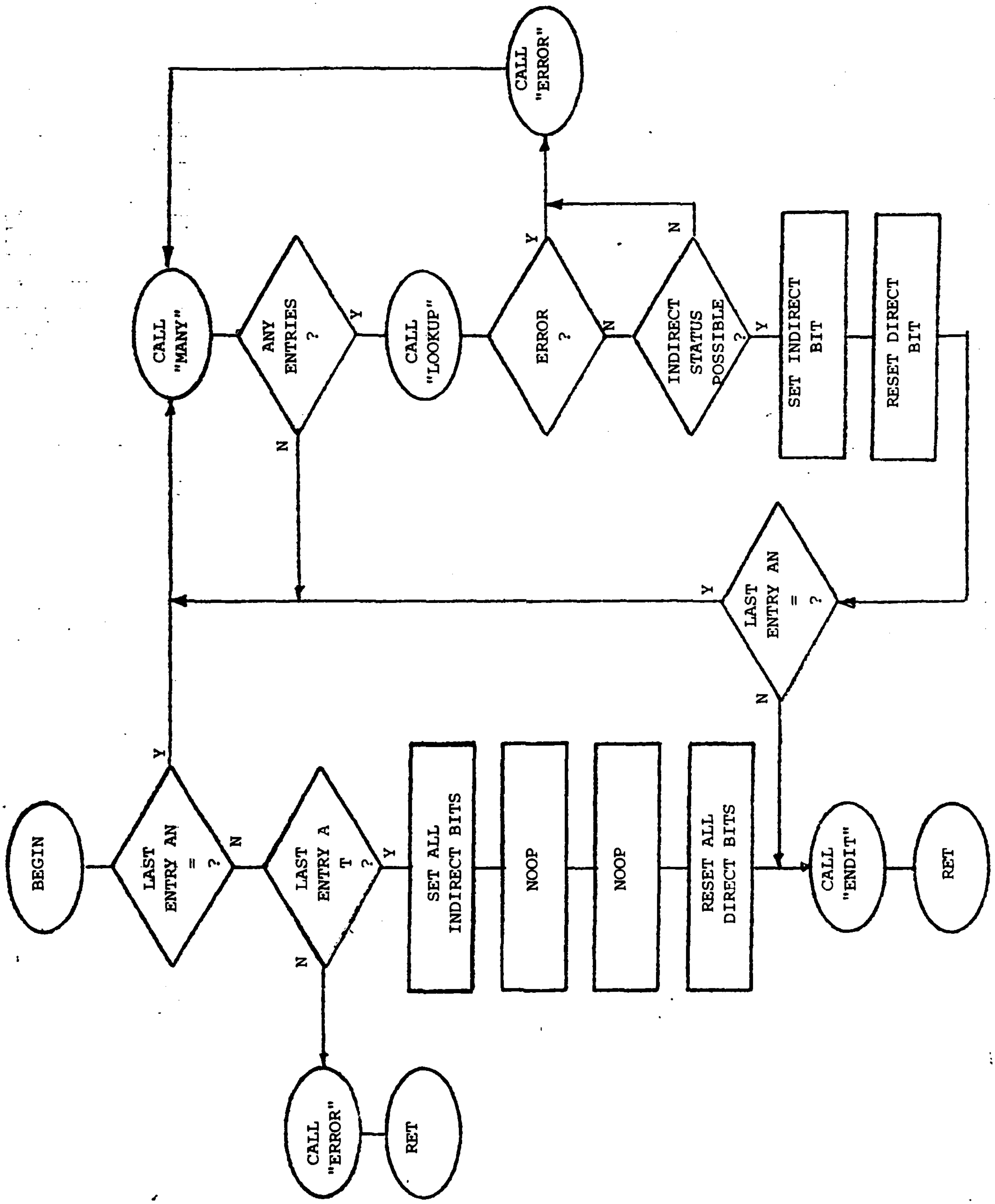

BEGIN
GET FUNCTION (F) FROM LOOKUP TABLE
F FROM LOOKUP = F IN DISPLAY BUFFER ?
Y
GET ADDRESS OF THE FUNCTION
GET STATUS ADDRESS OF THE FUNCTION
RET
N
GCU ACTIVE ?
N
Y
MORE THAN ONE CHARACTER IN DISPLAY BUFFER ?
Y
CALL "ERROR"
SET ERROR FLAG
RET
N
GET SINGLE KEY STOKE FUNCTION DEFINITION FROM LOOKUP TABLE
EQUAL TO CHARACT. IN ALPHANUM. DISPLAY BUFFER ?
Y
DISPLAY THE FUNCTION
N
POINT TO NEXT F IN LOOKUP TABLE
END ?
Y
N
CALL "ERROR"
RET

"LOOKUP"

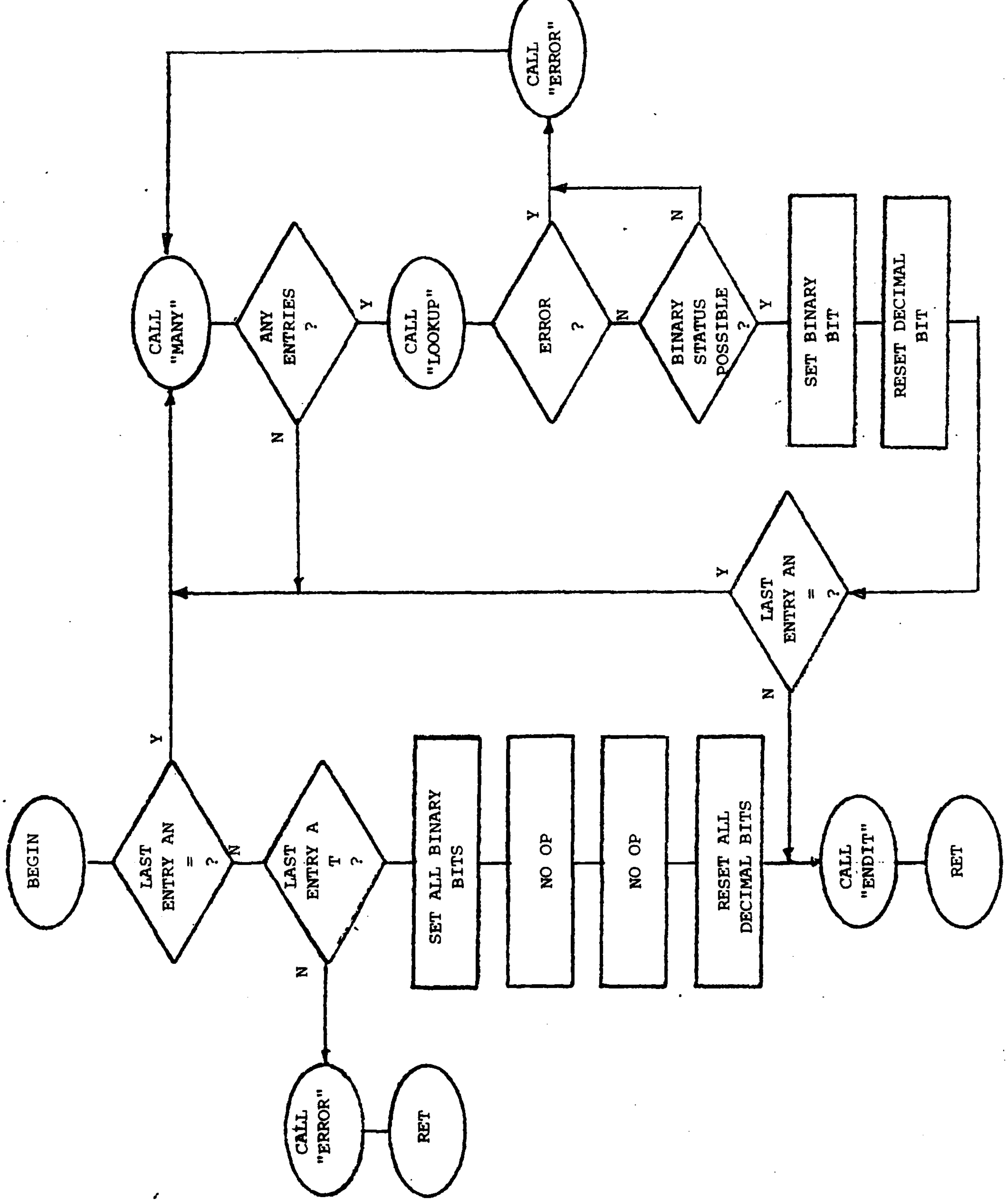

BEGIN
CALL "CLEAR"
CALL "NMREAD"
N
CR ?
Y
FIND NO OF ENTRIES IN HEX DIAPLAY BUFFER
NO OF ENTRIES > 4 ?
Y
CALL "CLEAR"
BUZZ
N
NO OF ENTRIES = 1 ?
Y
COMPACT 1 INTO LOWER BYTE
SET HI-BYTE =0
N
NO OF ENTRIES = 2 ?
Y
COMPACT 2 INTO LOWER BYTE
SET HI-BYTE =0
N
NO OF ENTRIES =3 ?
Y
COMPACT 3 INTO A WORD
N
COMPACT 4 INTO A WORD
A
"ADDRES"


A
GET THE MODE OF THE FUNCTION
DIRECT ACCESS ?
Y
N
INDIRECT ACCESS ?
N
Y
MODE?
BUZZ
SET ERROR FLAG
RET
BINARY INFO ?
N
DECIMAL INFO ?
N
Y
CALL "DECBIN"
ADDRESS FLAG SET ?
N
Y
ADD 2000H TO ADDRESS
RESET ERROR FLAG
RET

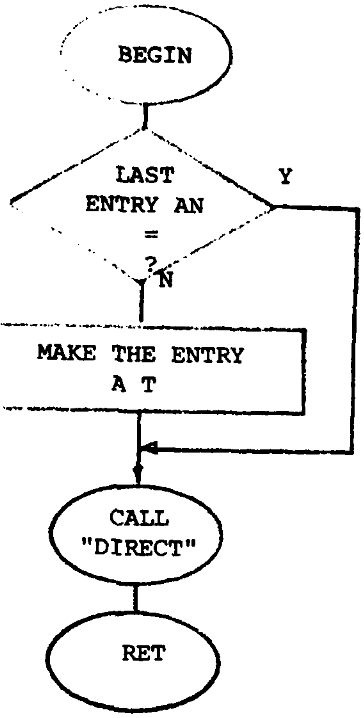
"ADDRES" CONT.


BEGIN
LAST ENTRY AN = ?
Y
N
LAST ENTRY A T ?
Y
N
CALL "ERROR"
RET
SET ALL INDIRECT BITS
NOOP
NOOP
RESET ALL DIRECT BITS
CALL "ENDIT"
RET
CALL "MANY"
ANY ENTRIES ?
Y
N
CALL "LOOKUP"
ERROR ?
Y
N
CALL "ERROR"
INDIRECT STATUS POSSIBLE ?
Y
N
SET INDIRECT BIT
RESET DIRECT BIT
LAST ENTRY AN = ?
Y
N

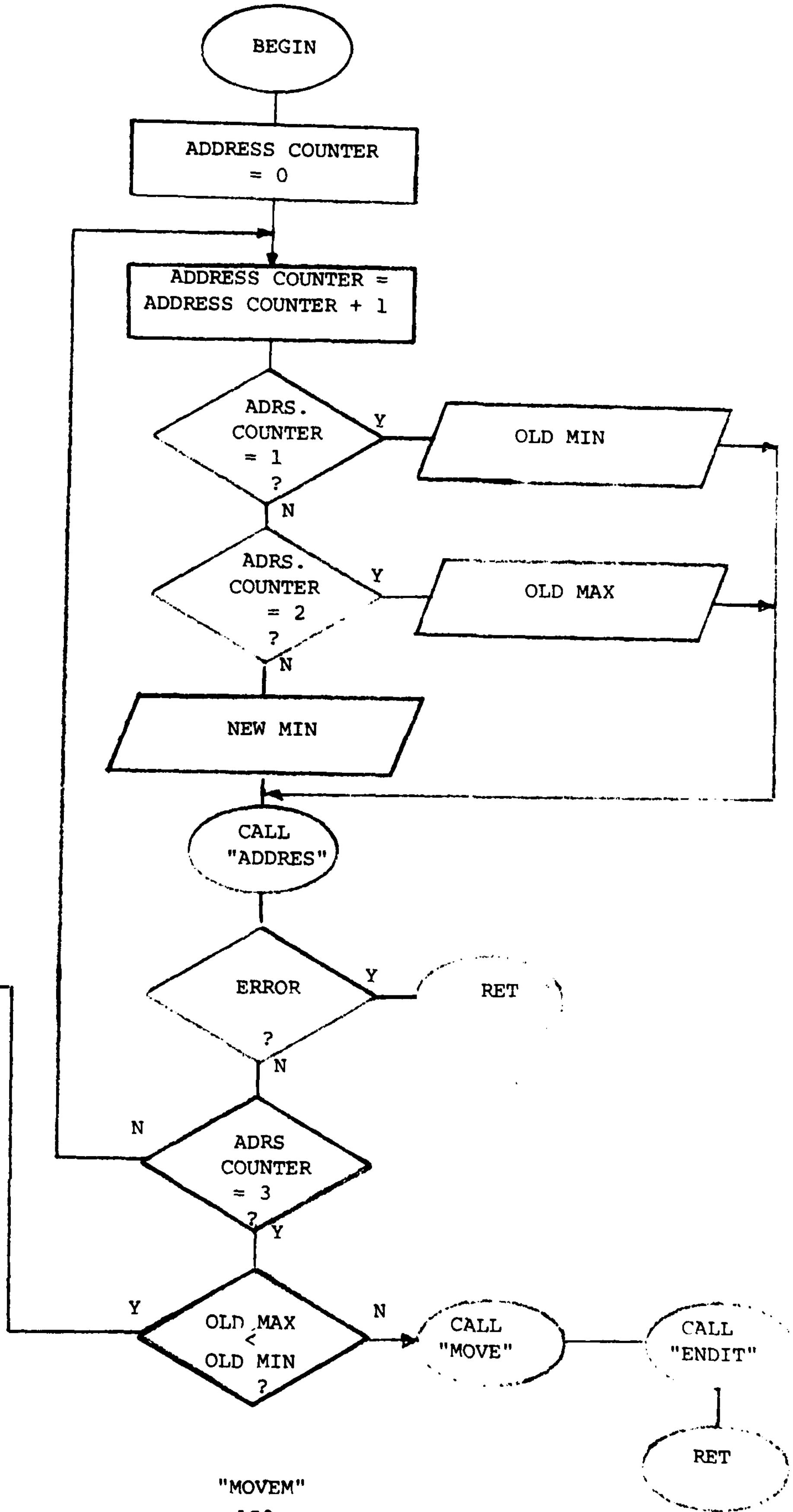
"INDRCT"

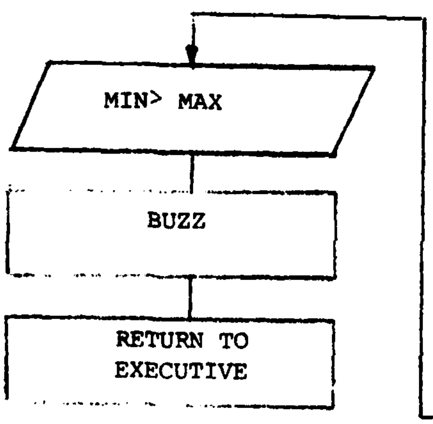

BEGIN
LAST ENTRY AN = ?
Y
N
CALL "MANY"
ANY ENTRIES ?
Y
N
CALL "LOOKUP"
ERROR ?
Y
N
CALL "ERROR"
BINARY STATUS POSSIBLE ?
Y
N
SET BINARY BIT
RESET DECIMAL BIT
LAST ENTRY AN = ?
Y
N
LAST ENTRY A T ?
N
CALL "ERROR"
RET
SET ALL BINARY BITS
NO OP
NO OP
RESET ALL DECIMAL BITS
CALL "ENDIT"
RET

"BNRY"

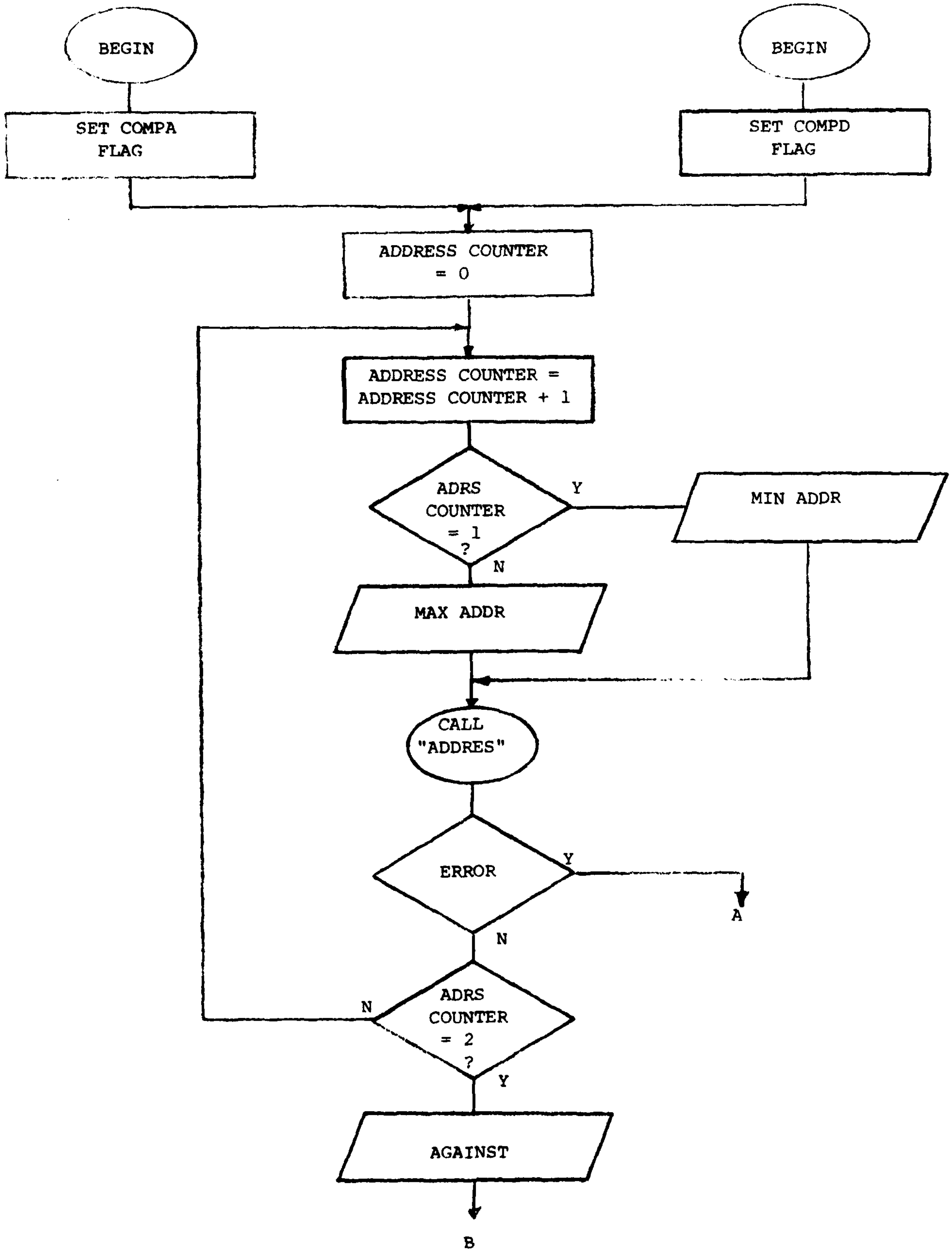

BEGIN
LAST ENTRY AN = ?
Y
N
MAKE THE ENTRY A T
CALL "DIRECT"
RET

"DCML"

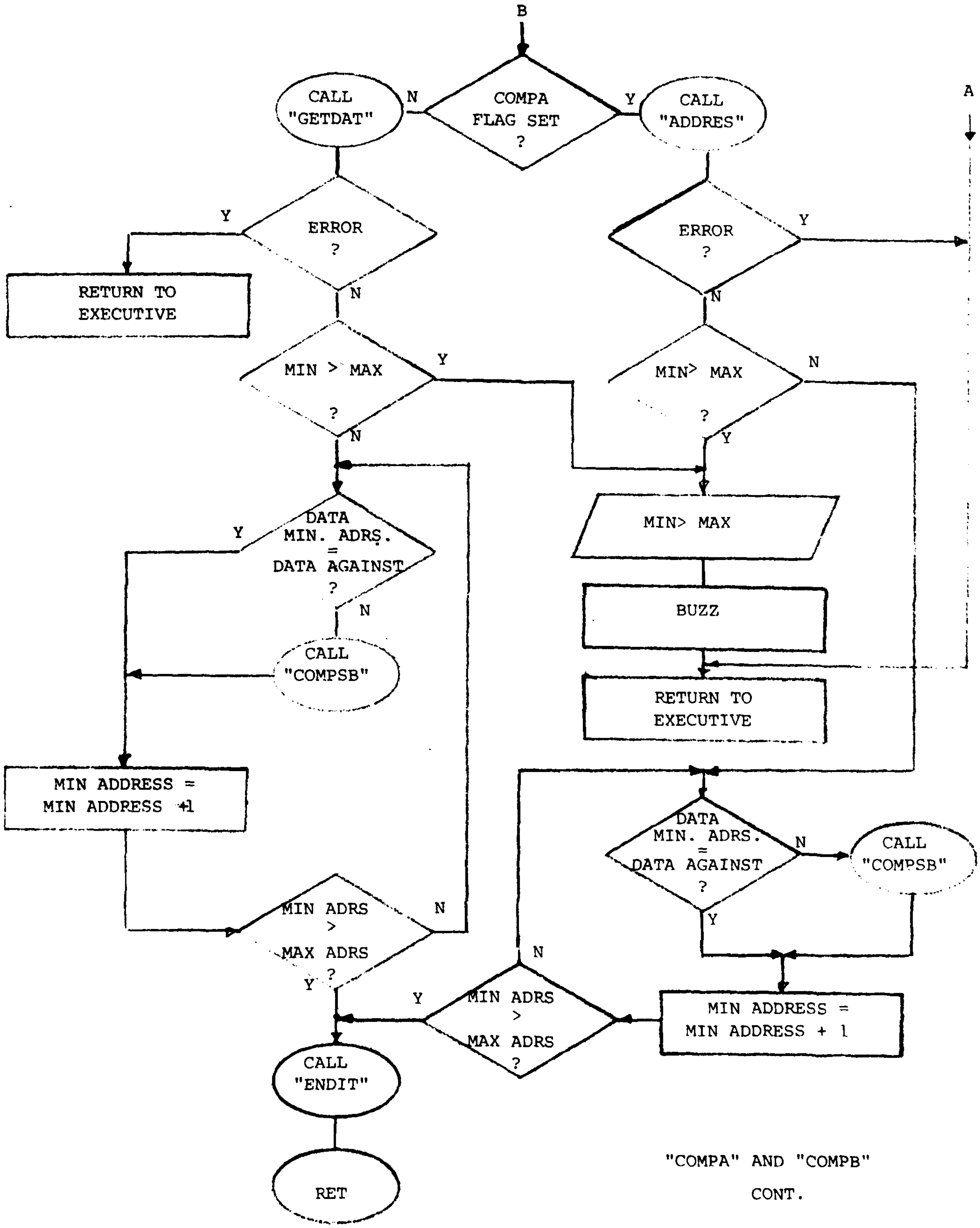

BEGIN
ADDRESS COUNTER = 0
ADDRESS COUNTER = ADDRESS COUNTER + 1
ADRS. COUNTER = 1 ?
Y
N
OLD MIN
ADRS. COUNTER = 2 ?
Y
N
OLD MAX
NEW MIN
CALL "ADDRES"
ERROR ?
Y
N
RET
ADRS COUNTER = 3 ?
N
Y
OLD MAX < OLD MIN ?
Y
N
CALL "MOVE"
CALL "ENDIT"
RET


MIN> MAX
BUZZ
RETURN TO EXECUTIVE

"MOVEM"

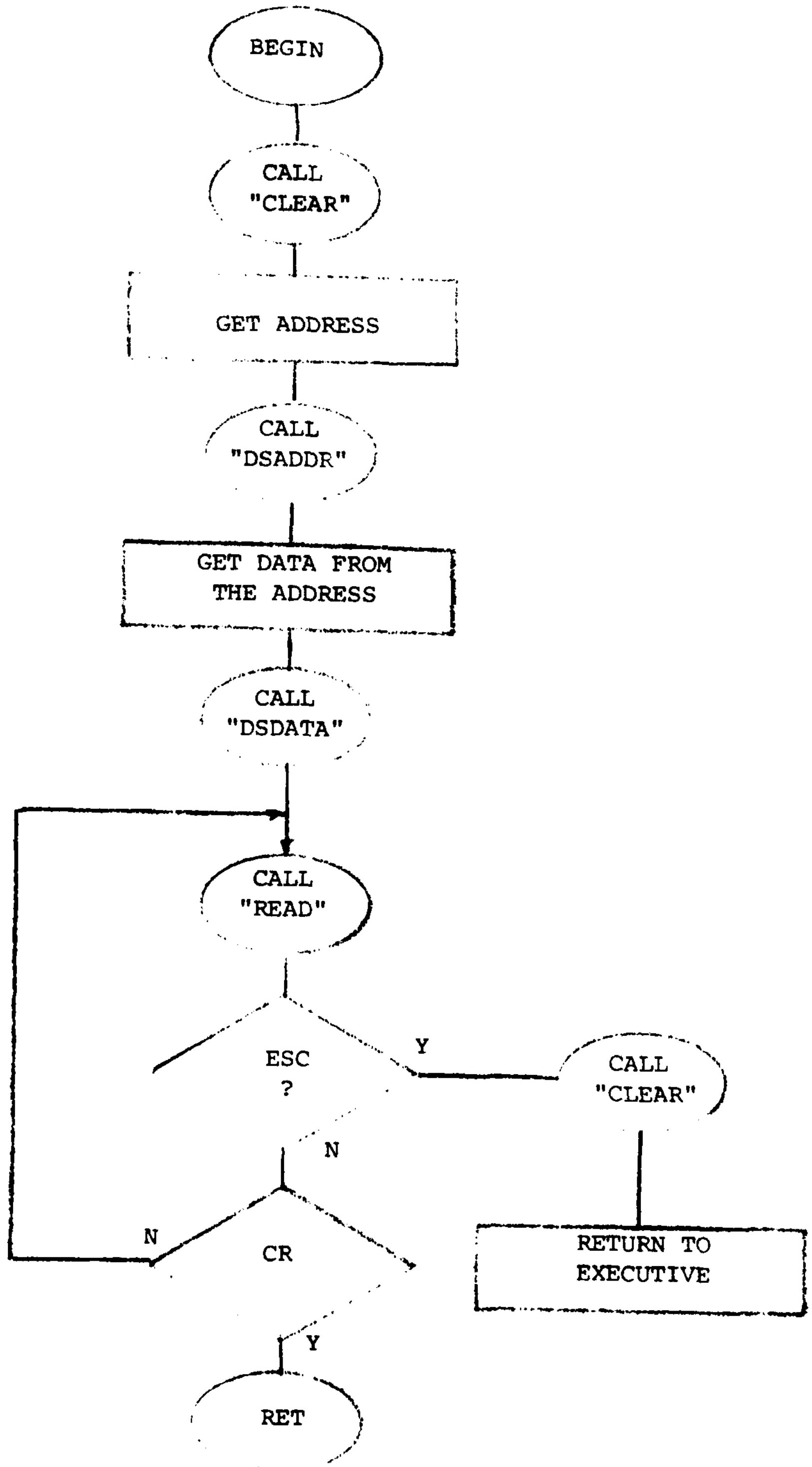


"COMPA" AND COMPD"

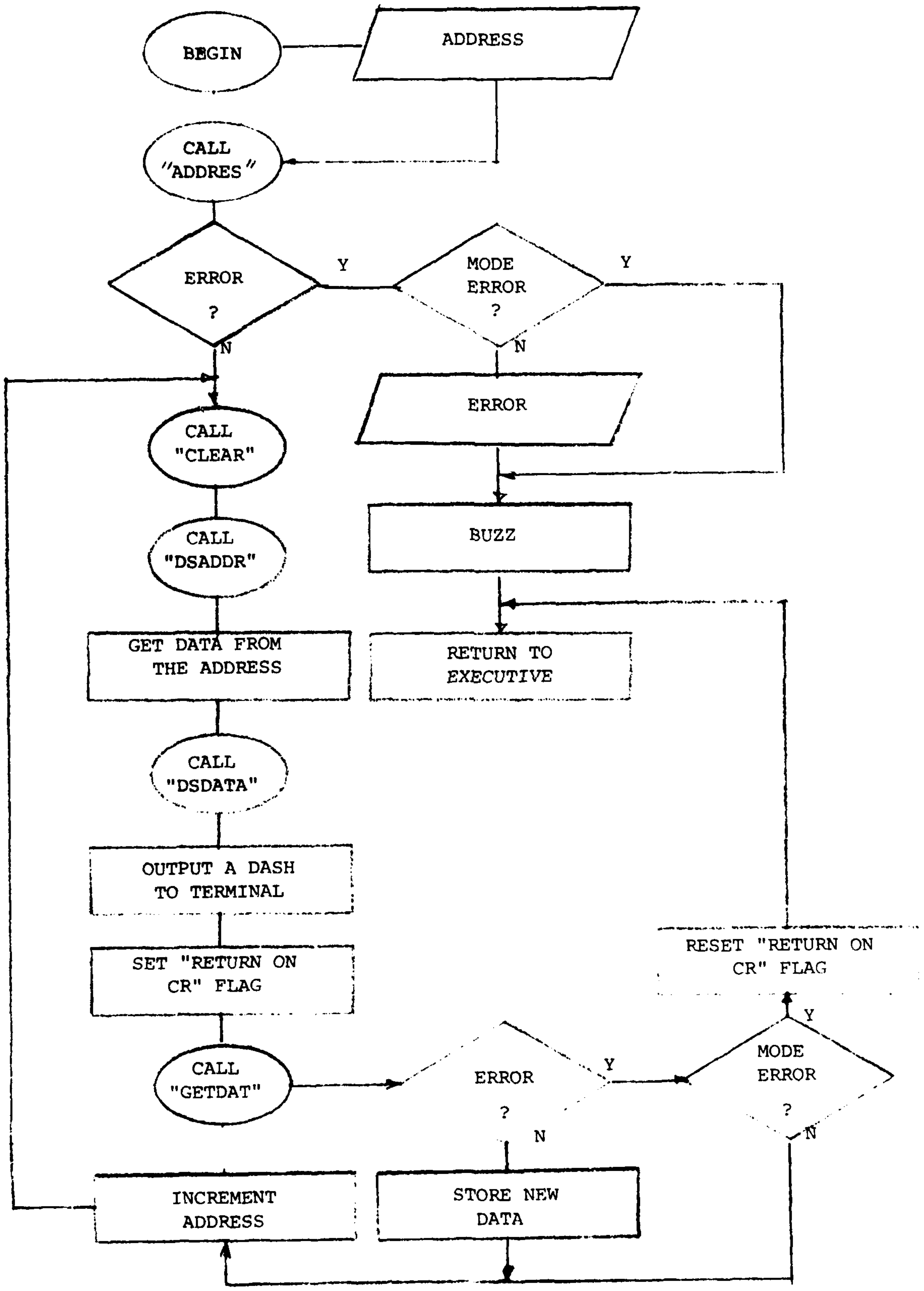


"COMPA" AND "COMPB"

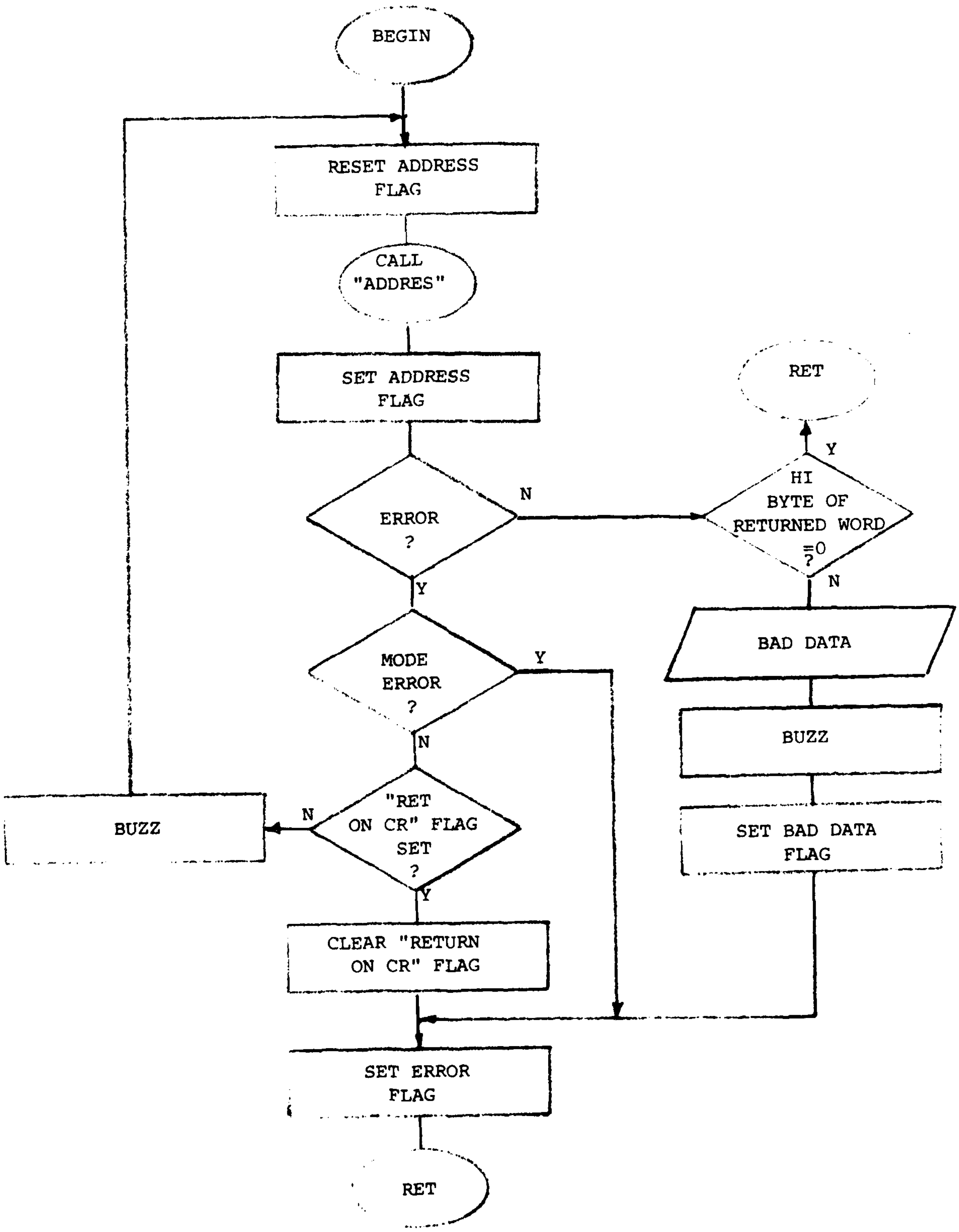
CONT.


BEGIN
CALL "CLEAR"
GET ADDRESS
CALL "DSADDR"
GET DATA FROM THE ADDRESS
CALL "DSDATA"
CALL "READ"
ESC ?
Y
CALL "CLEAR"
RETURN TO EXECUTIVE
N
CR
N
Y
RET

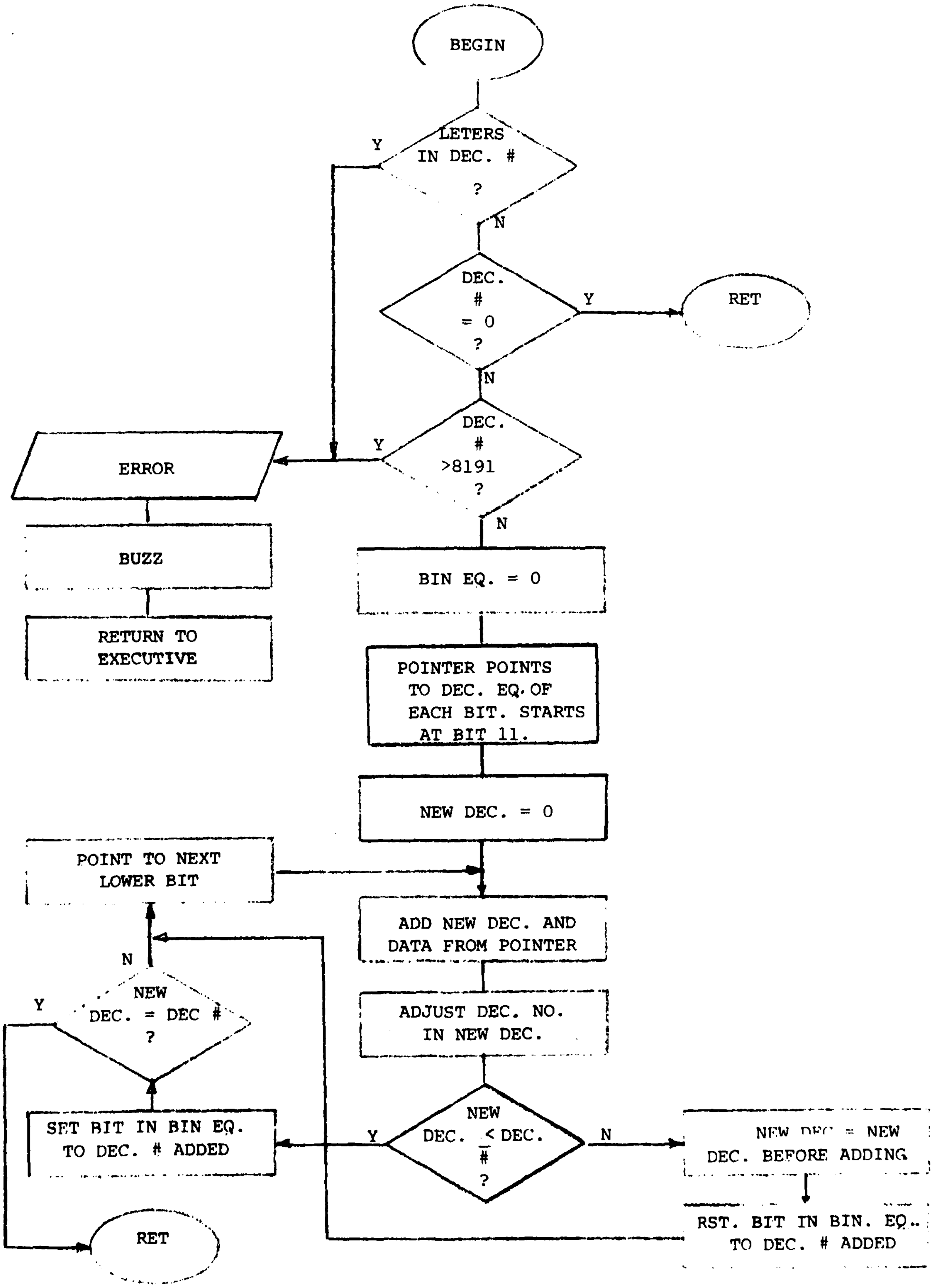
"COMPSB"


BEGIN
ADDRESS
CALL "ADDRES"
ERROR ?
Y
N
MODE ERROR ?
Y
N
ERROR
CALL "CLEAR"
BUZZ
CALL "DSADDR"
GET DATA FROM THE ADDRESS
RETURN TO EXECUTIVE
CALL "DSDATA"
OUTPUT A DASH TO TERMINAL
RESET "RETURN ON CR" FLAG
SET "RETURN ON CR" FLAG
Y
CALL "GETDAT"
ERROR ?
Y
N
MODE ERROR ?
N
INCREMENT ADDRESS
STORE NEW DATA

"ALTR"

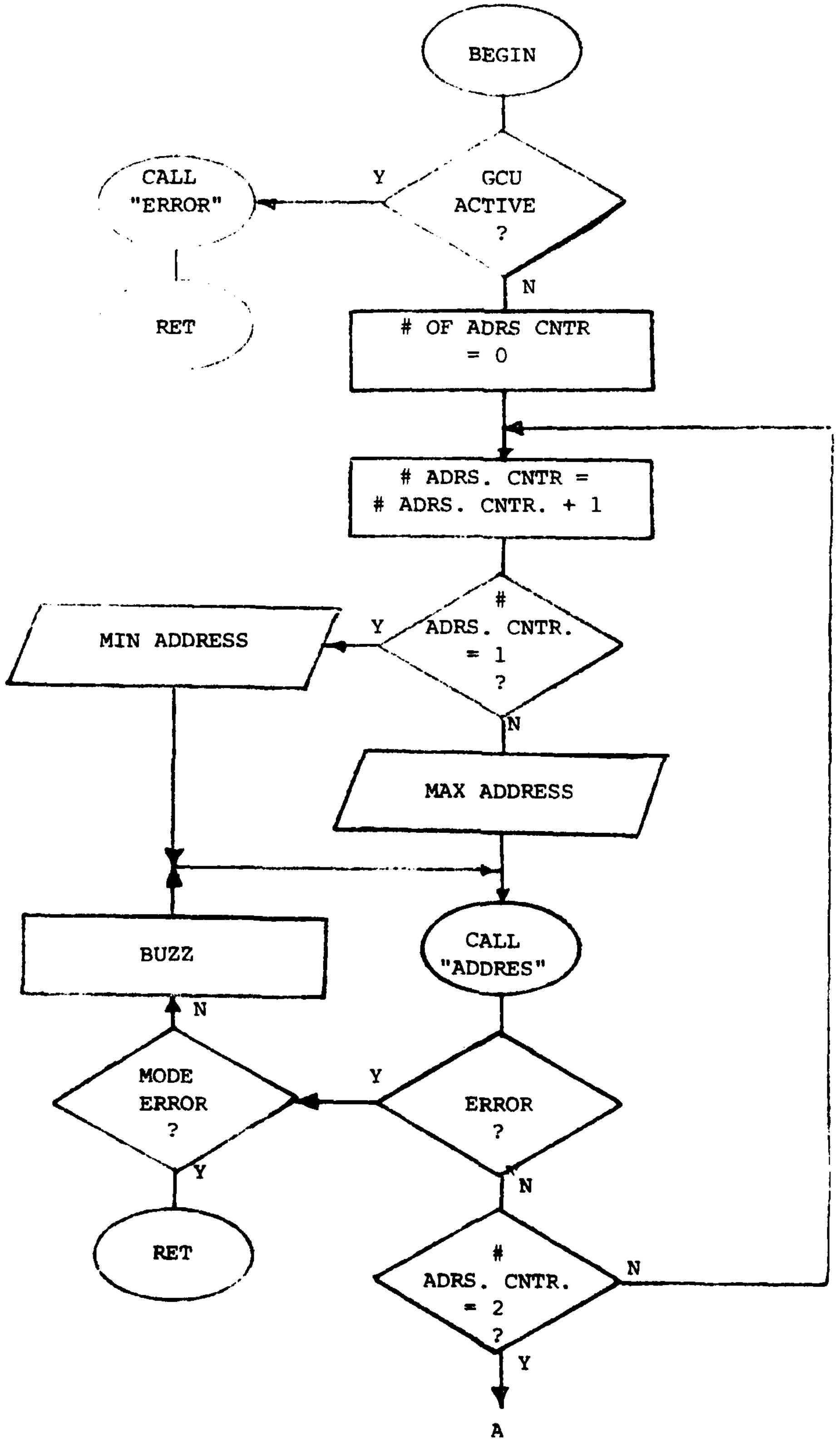

BEGIN
RESET ADDRESS FLAG
CALL "ADDRES"
SET ADDRESS FLAG
ERROR ?
N
Y
HI BYTE OF RETURNED WORD =0 ?
Y
RET
N
BAD DATA
BUZZ
SET BAD DATA FLAG
MODE ERROR ?
Y
N
"RET ON CR" FLAG SET ?
N
BUZZ
Y
CLEAR "RETURN ON CR" FLAG
SET ERROR FLAG
RET

"GETDAT"

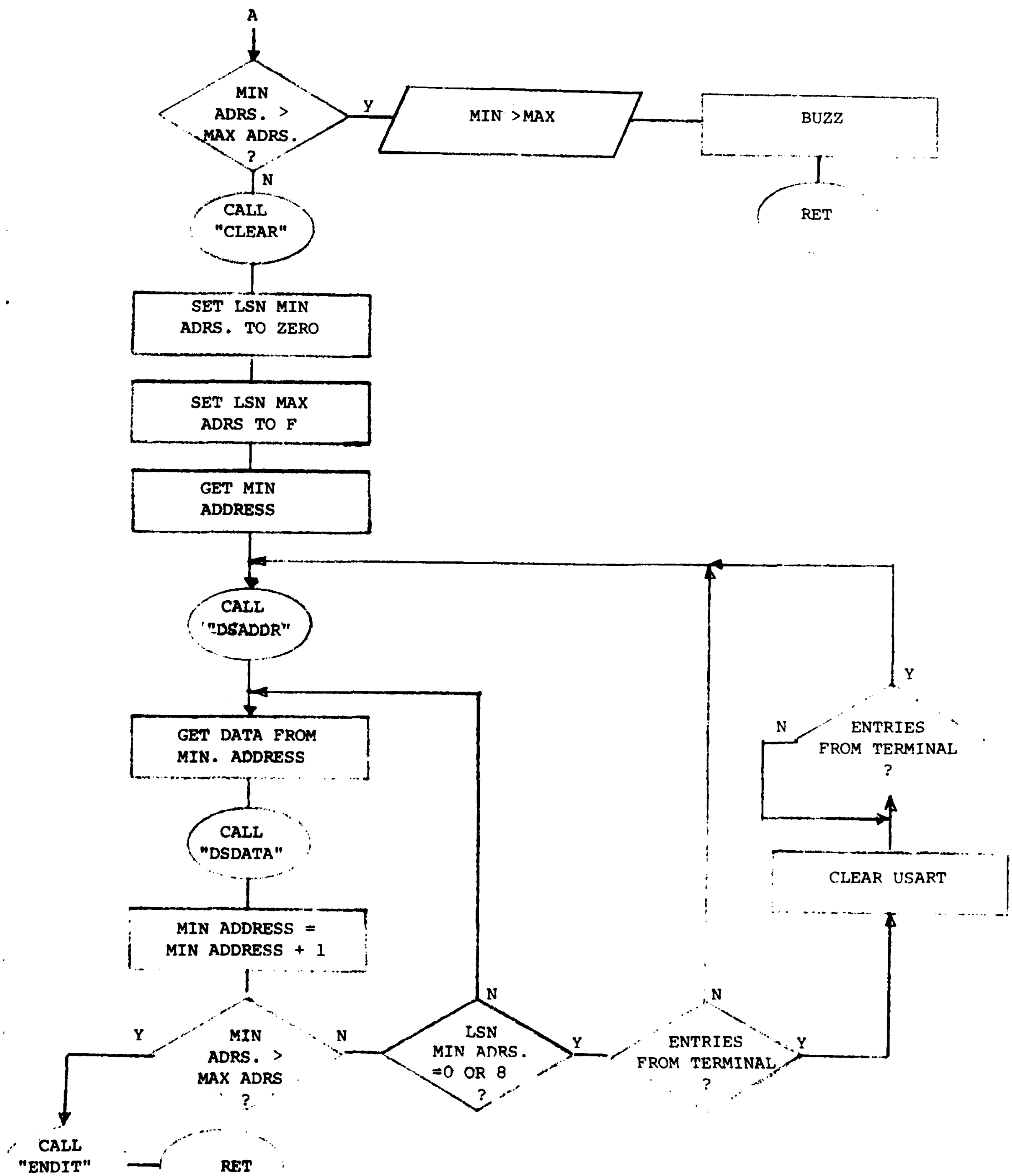


BEGIN
LETERS IN DEC. # ?
Y
N
DEC. # = 0 ?
Y
RET
N
DEC. # >8191 ?
Y
ERROR
BUZZ
RETURN TO EXECUTIVE
N
BIN EQ. = 0
POINTER POINTS TO DEC. EQ. OF EACH BIT. STARTS AT BIT 11.
NEW DEC. = 0
POINT TO NEXT LOWER BIT
ADD NEW DEC. AND DATA FROM POINTER
ADJUST DEC. NO. IN NEW DEC.
N
NEW DEC. = DEC # ?
Y
NEW DEC. ≤ DEC. # ?
Y
N
SET BIT IN BIN EQ. TO DEC. # ADDED
NEW DEC = NEW DEC. BEFORE ADDING
RST. BIT IN BIN. EQ. TO DEC. # ADDED
RET

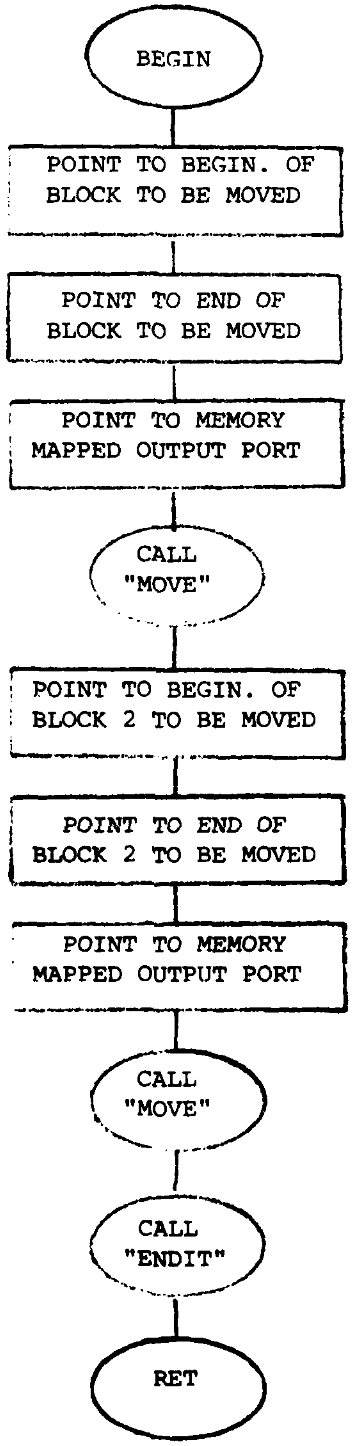
"DECBIN"


BEGIN
GCU ACTIVE ?
Y
CALL "ERROR"
RET
N
OF ADRS CNTR = 0
ADRS. CNTR = # ADRS. CNTR. + 1
ADRS. CNTR. = 1 ?
Y
MIN ADDRESS
N
MAX ADDRESS
CALL "ADDRES"
BUZZ
N
MODE ERROR ?
Y
ERROR ?
Y
RET
N
ADRS. CNTR. = 2 ?
N
Y
A

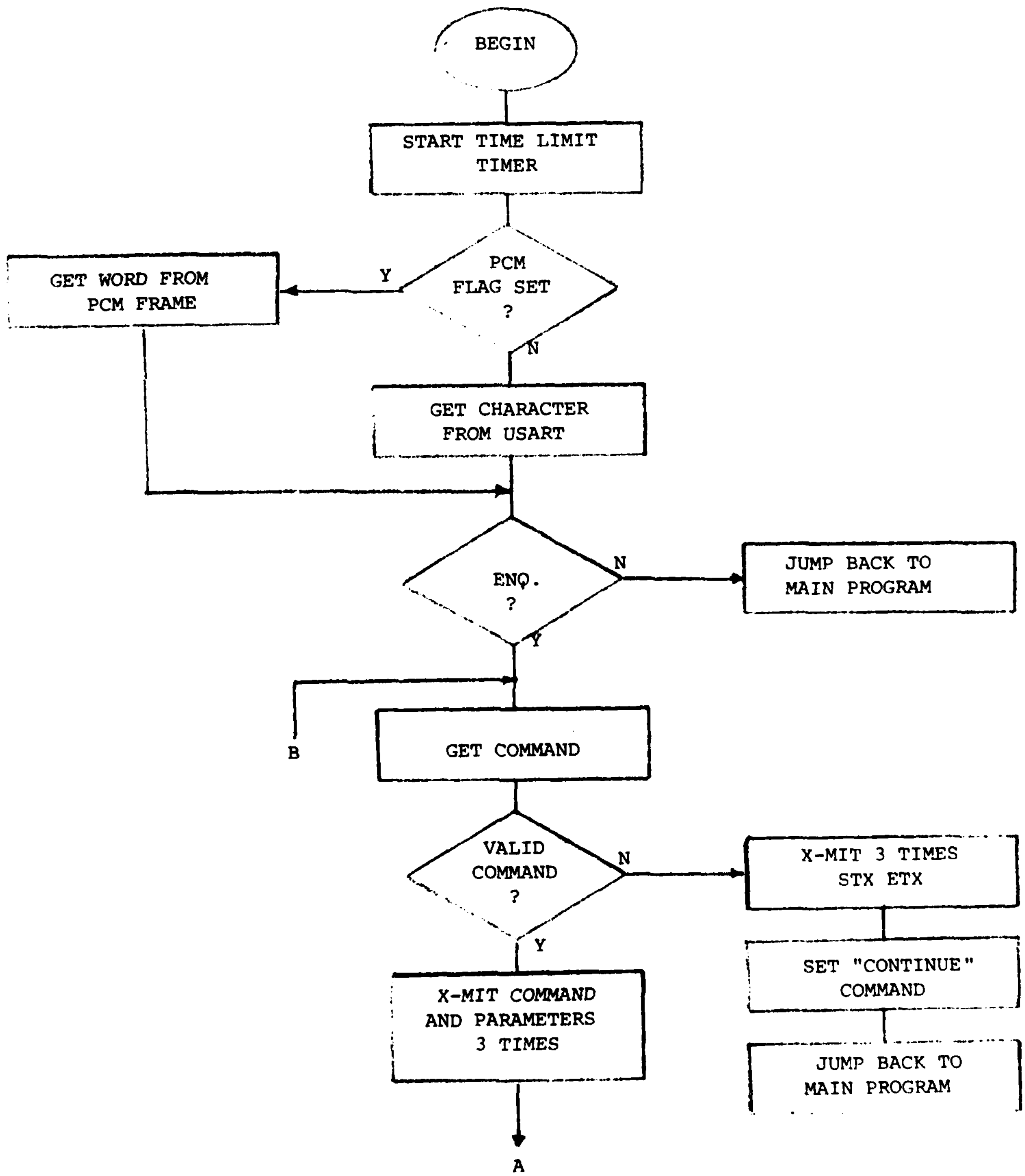
"DISPL"


A
MIN ADRS. > MAX ADRS. ?
Y
N
MIN >MAX
BUZZ
RET
CALL "CLEAR"
SET LSN MIN ADRS. TO ZERO
SET LSN MAX ADRS TO F
GET MIN ADDRESS
CALL "DSADDR"
GET DATA FROM MIN. ADDRESS
CALL "DSDATA"
MIN ADDRESS = MIN ADDRESS + 1
MIN ADRS. > MAX ADRS ?
Y
N
LSN MIN ADRS. =0 OR 8 ?
N
Y
ENTRIES FROM TERMINAL ?
N
Y
CLEAR USART
ENTRIES FROM TERMINAL ?
N
Y
CALL "ENDIT"
RET

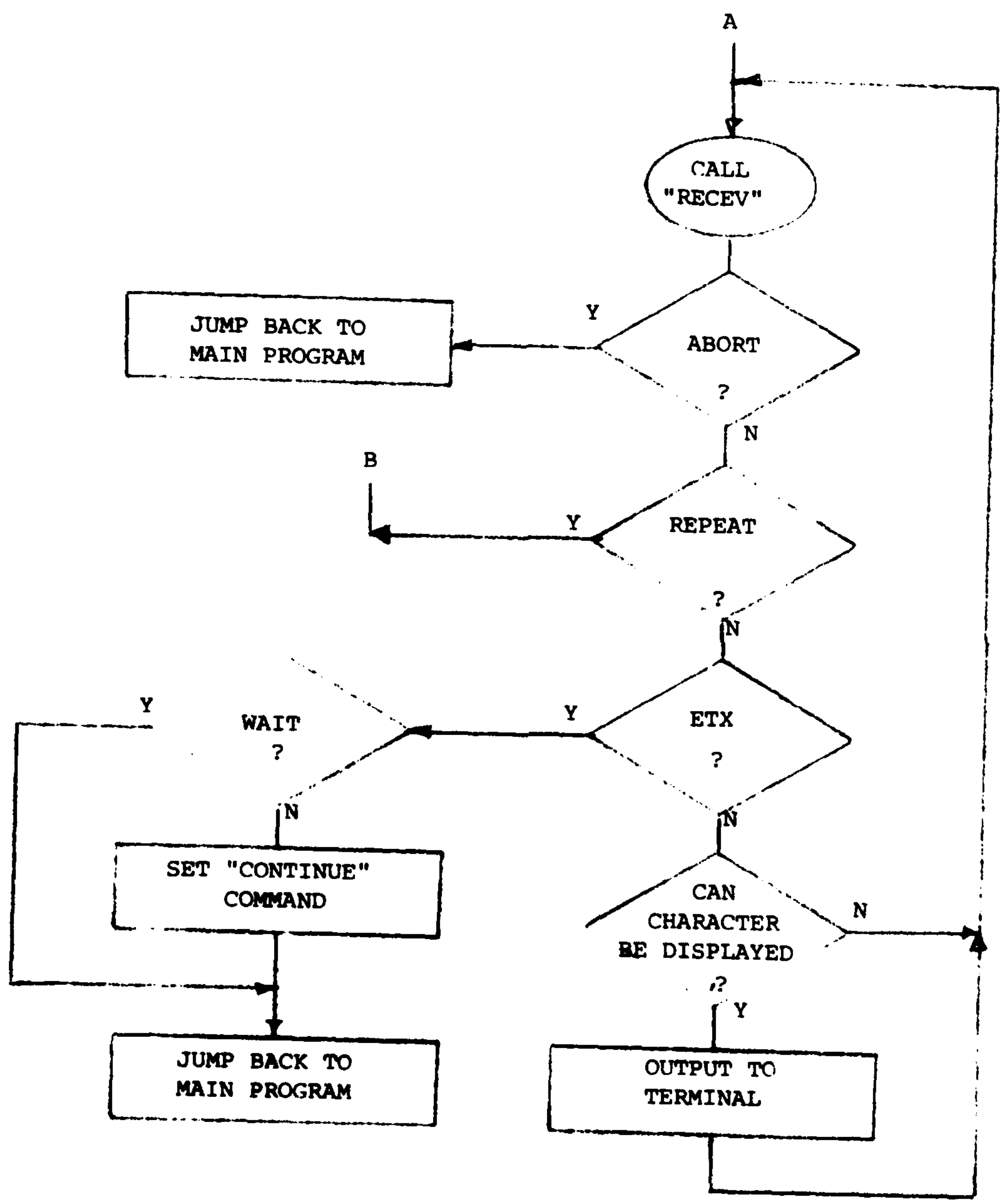
"DISPL" CONT.


BEGIN
POINT TO BEGIN. OF BLOCK TO BE MOVED
POINT TO END OF BLOCK TO BE MOVED
POINT TO MEMORY MAPPED OUTPUT PORT
CALL "MOVE"
POINT TO BEGIN. OF BLOCK 2 TO BE MOVED
POINT TO END OF BLOCK 2 TO BE MOVED
POINT TO MEMORY MAPPED OUTPUT PORT
CALL "MOVE"
CALL "ENDIT"
RET

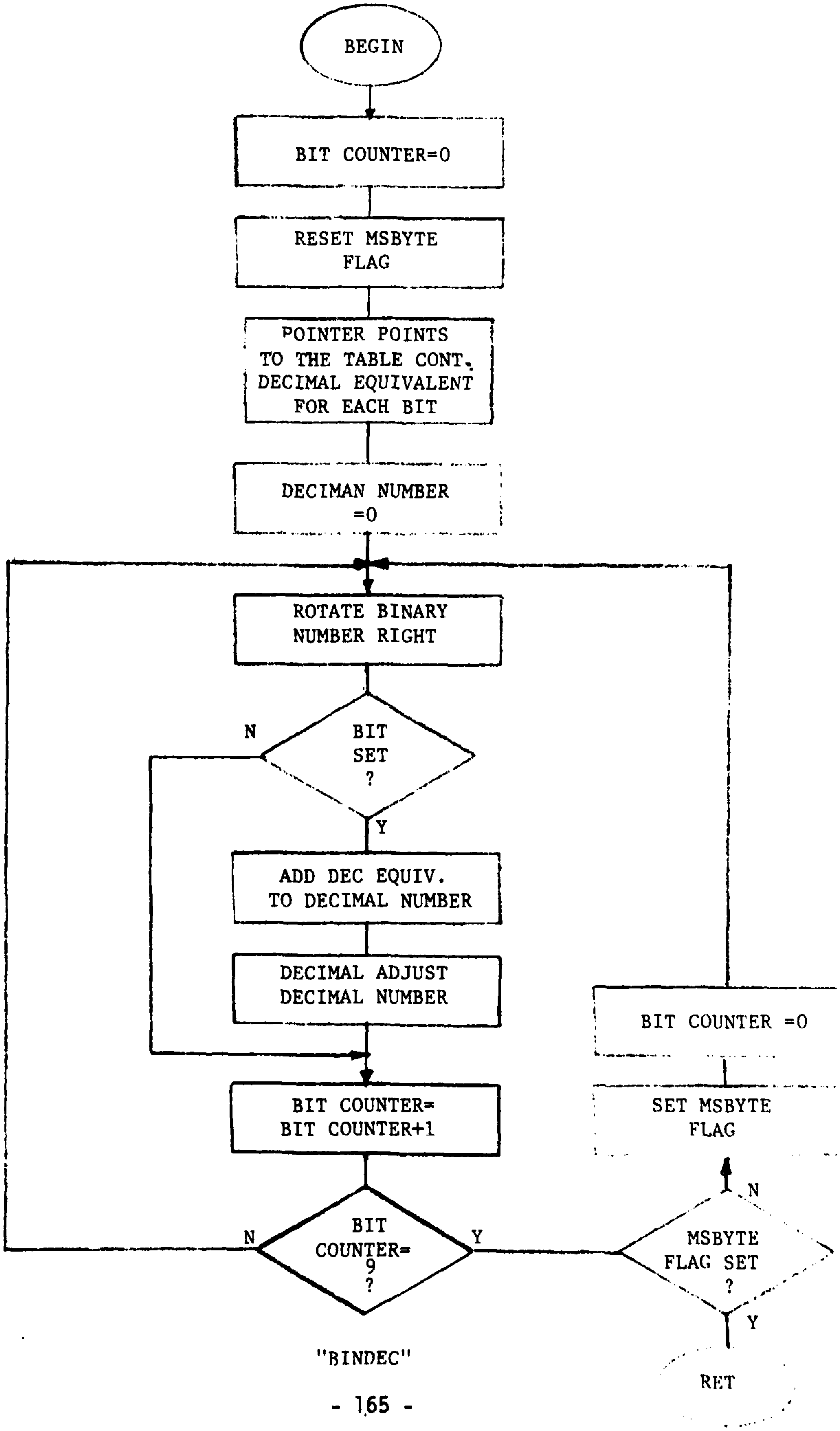
"FEPROM"


BEGIN
START TIME LIMIT TIMER
PCM FLAG SET ?
Y
N
GET WORD FROM PCM FRAME
GET CHARACTER FROM USART
ENQ. ?
N
Y
JUMP BACK TO MAIN PROGRAM
B
GET COMMAND
VALID COMMAND ?
N
Y
X-MIT 3 TIMES STX ETX
SET "CONTINUE" COMMAND
JUMP BACK TO MAIN PROGRAM
X-MIT COMMAND AND PARAMETERS 3 TIMES
A

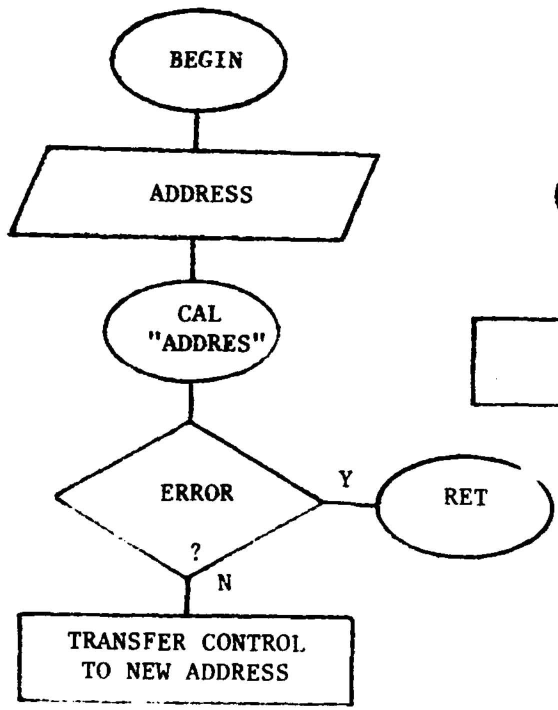
"TTYLNK"


A
CALL "RECEV"
JUMP BACK TO MAIN PROGRAM
Y
ABORT ?
N
B
Y
REPEAT ?
N
Y
WAIT ?
Y
ETX ?
N
N
SET "CONTINUE" COMMAND
CAN CHARACTER BE DISPLAYED ?
N
Y
JUMP BACK TO MAIN PROGRAM
OUTPUT TO TERMINAL

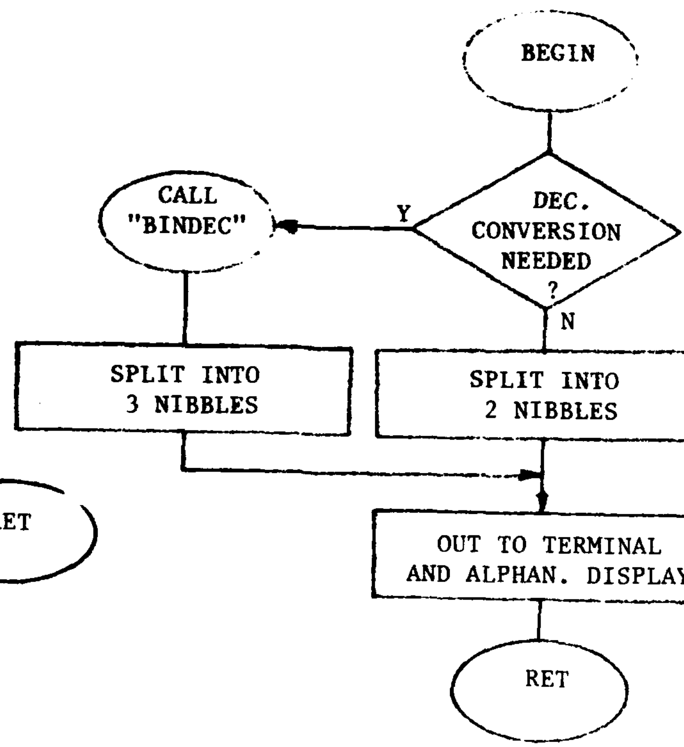
"TTY LNK " CONT.


BEGIN
BIT COUNTER=0
RESET MSBYTE FLAG
POINTER POINTS TO THE TABLE CONT. DECIMAL EQUIVALENT FOR EACH BIT
DECIMAN NUMBER =0
ROTATE BINARY NUMBER RIGHT
BIT SET ?
N
Y
ADD DEC EQUIV. TO DECIMAL NUMBER
DECIMAL ADJUST DECIMAL NUMBER
BIT COUNTER= BIT COUNTER+1
BIT COUNTER= 9 ?
N
Y
MSBYTE FLAG SET ?
N
Y
SET MSBYTE FLAG
BIT COUNTER =0
RET

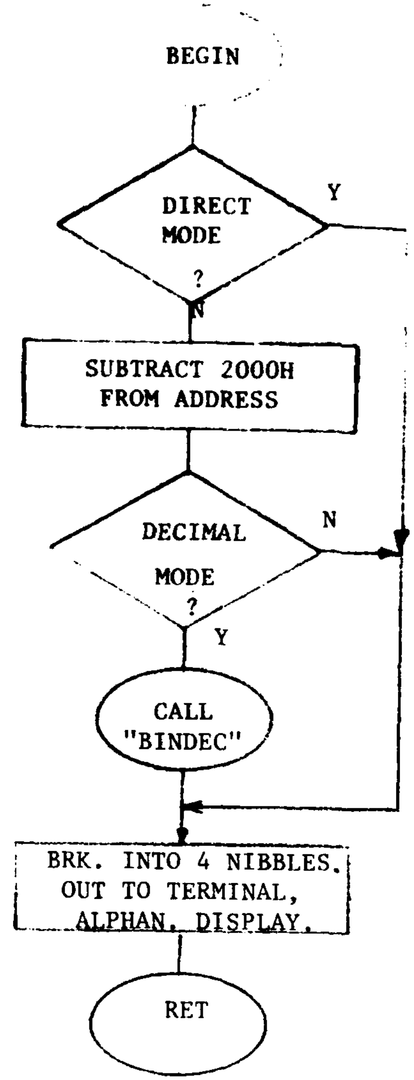
"BINDEC"



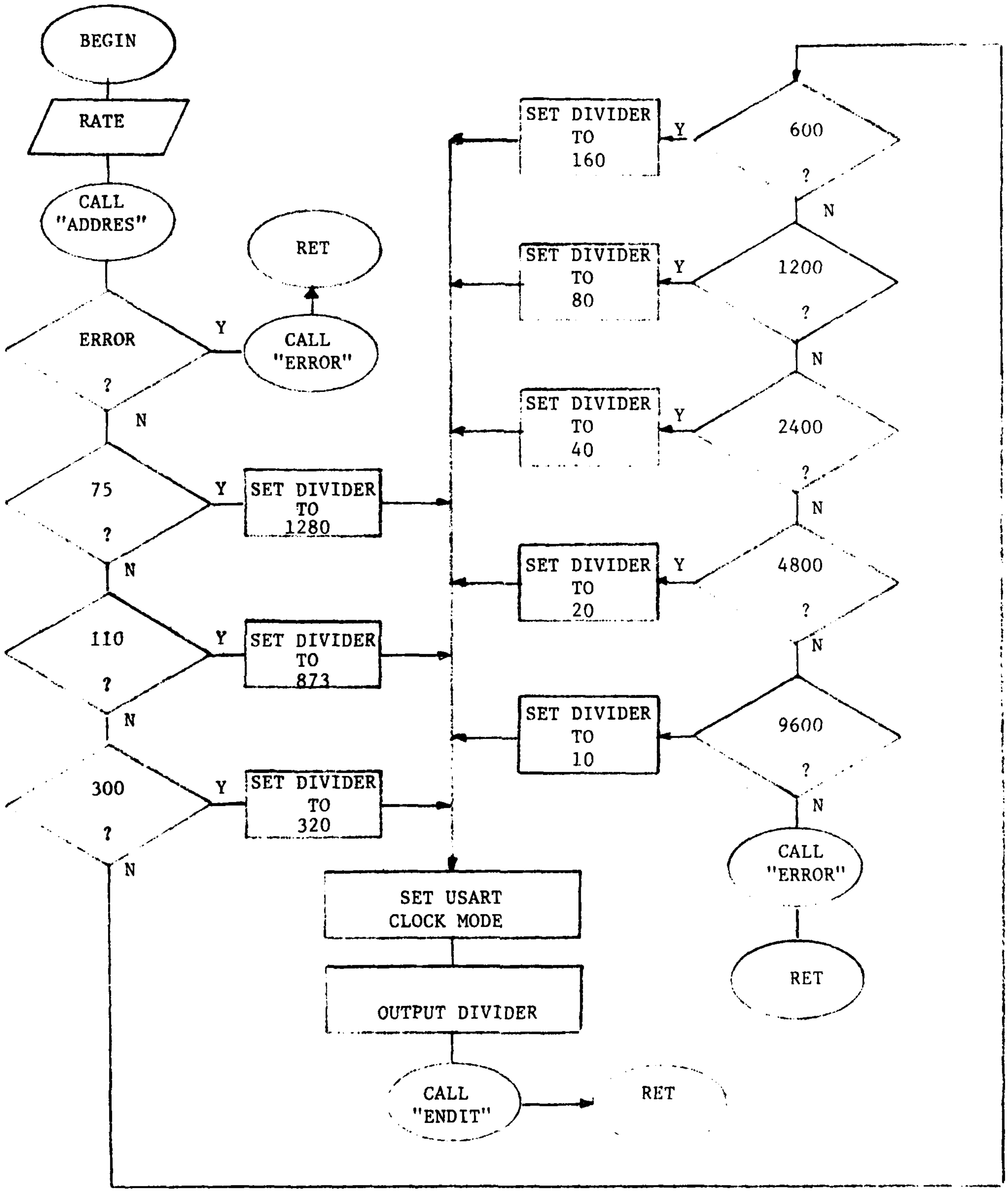
"GOTO"



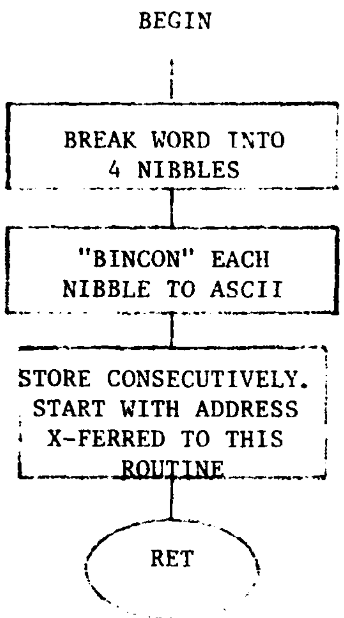
"DSDATA"



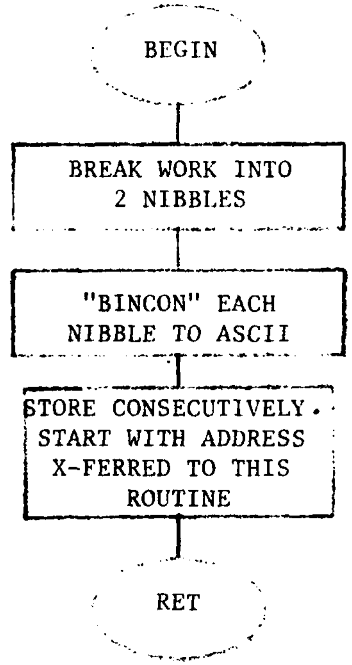
"DSDATA"


BEGIN
RATE
CALL "ADDRES"
ERROR ?
Y
N
CALL "ERROR"
RET
75 ?
SET DIVIDER TO 1280
110 ?
SET DIVIDER TO 873
300 ?
SET DIVIDER TO 320
600 ?
SET DIVIDER TO 160
1200 ?
SET DIVIDER TO 80
2400 ?
SET DIVIDER TO 40
4800 ?
SET DIVIDER TO 20
9600 ?
SET DIVIDER TO 10
CALL "ERROR"
RET
SET USART CLOCK MODE
OUTPUT DIVIDER
CALL "ENDIT"
RET

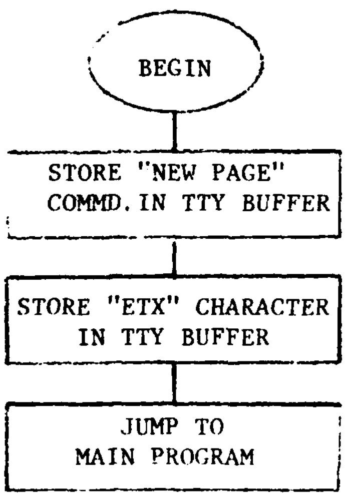
"BAUD"



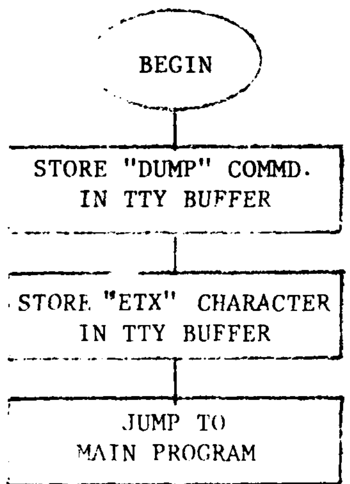
"CNVRT"



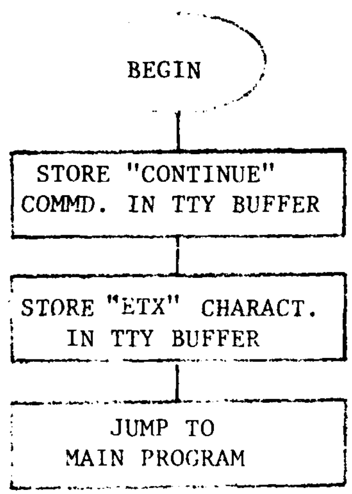
"CNVRT1"



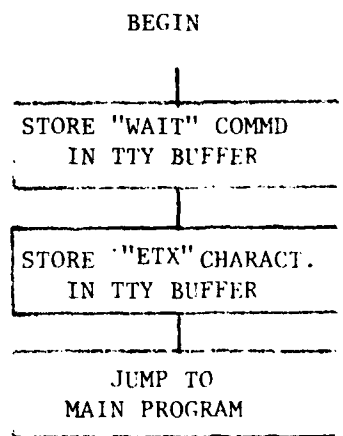
"NPAGE"



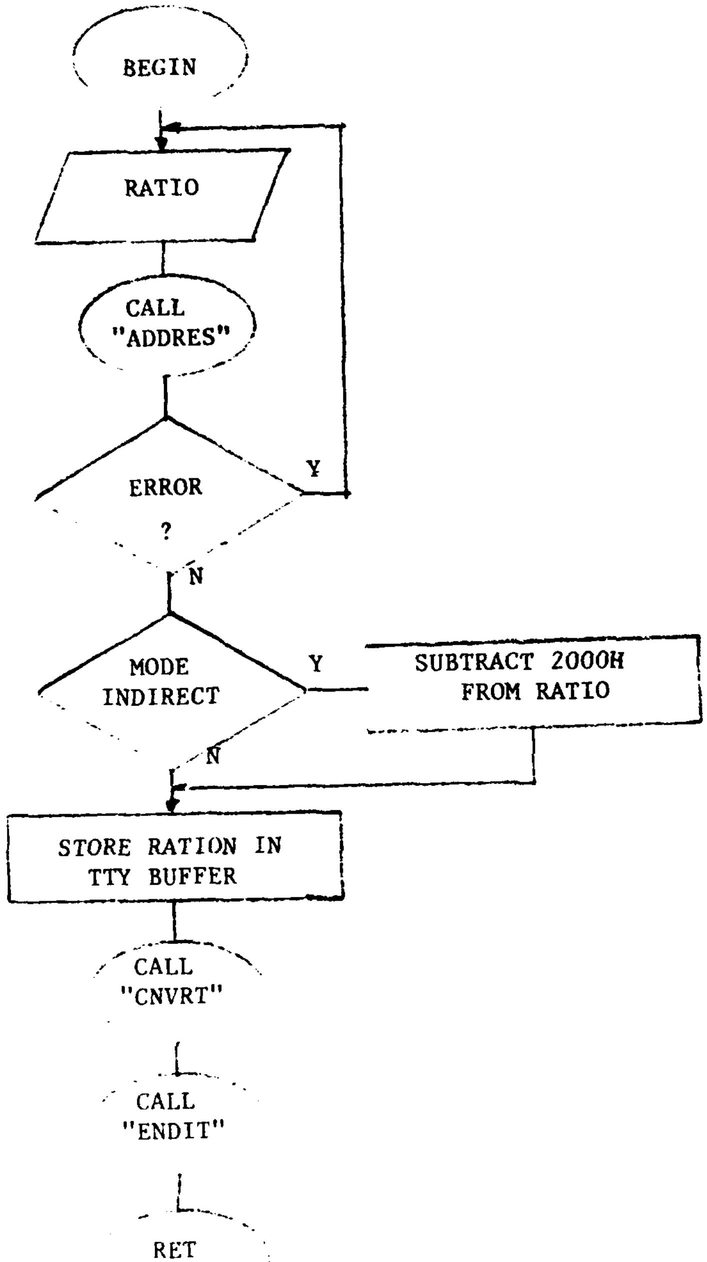
"DUMP"



"CONT."

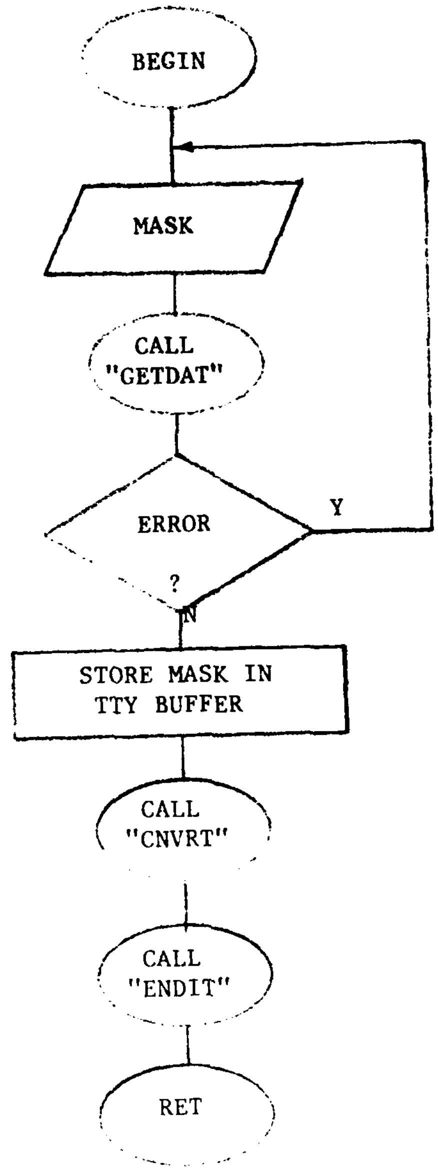


"WAIT"



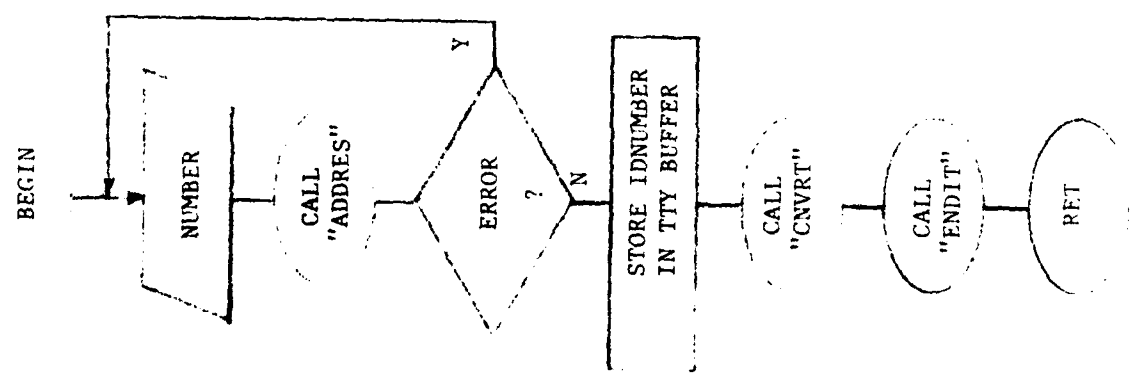
BEGIN
RATIO
CALL "ADDRES"
ERROR ?
Y
N
MODE INDIRECT
Y
SUBTRACT 2000H FROM RATIO
N
STORE RATION IN TTY BUFFER
CALL "CNVRT"
CALL "ENDIT"
RET

"RATIO"

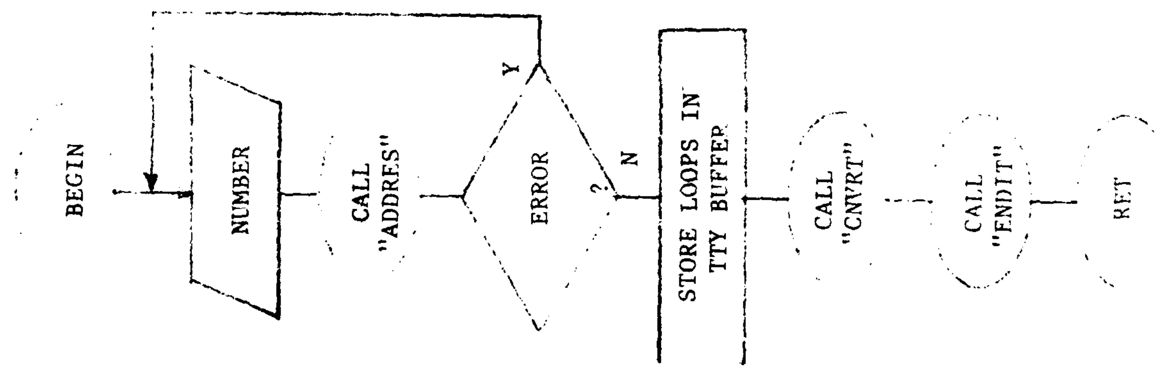


BEGIN
MASK
CALL "GETDAT"
ERROR ?
Y
N
STORE MASK IN TTY BUFFER
CALL "CNVRT"
CALL "ENDIT"
RET

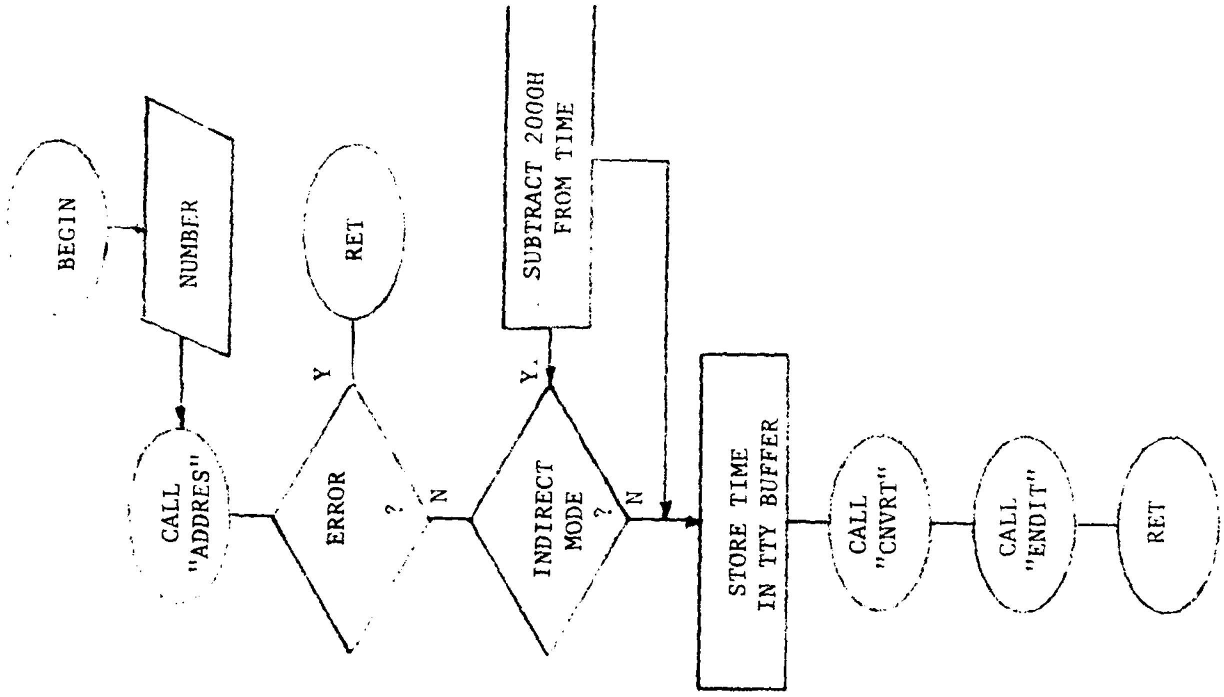
"MASK"



"IDNUM"



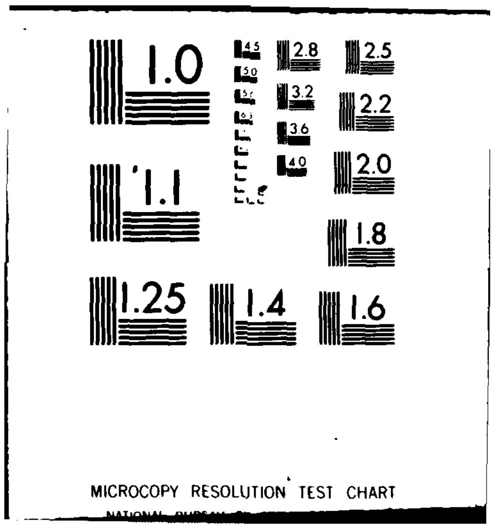
"LOP"

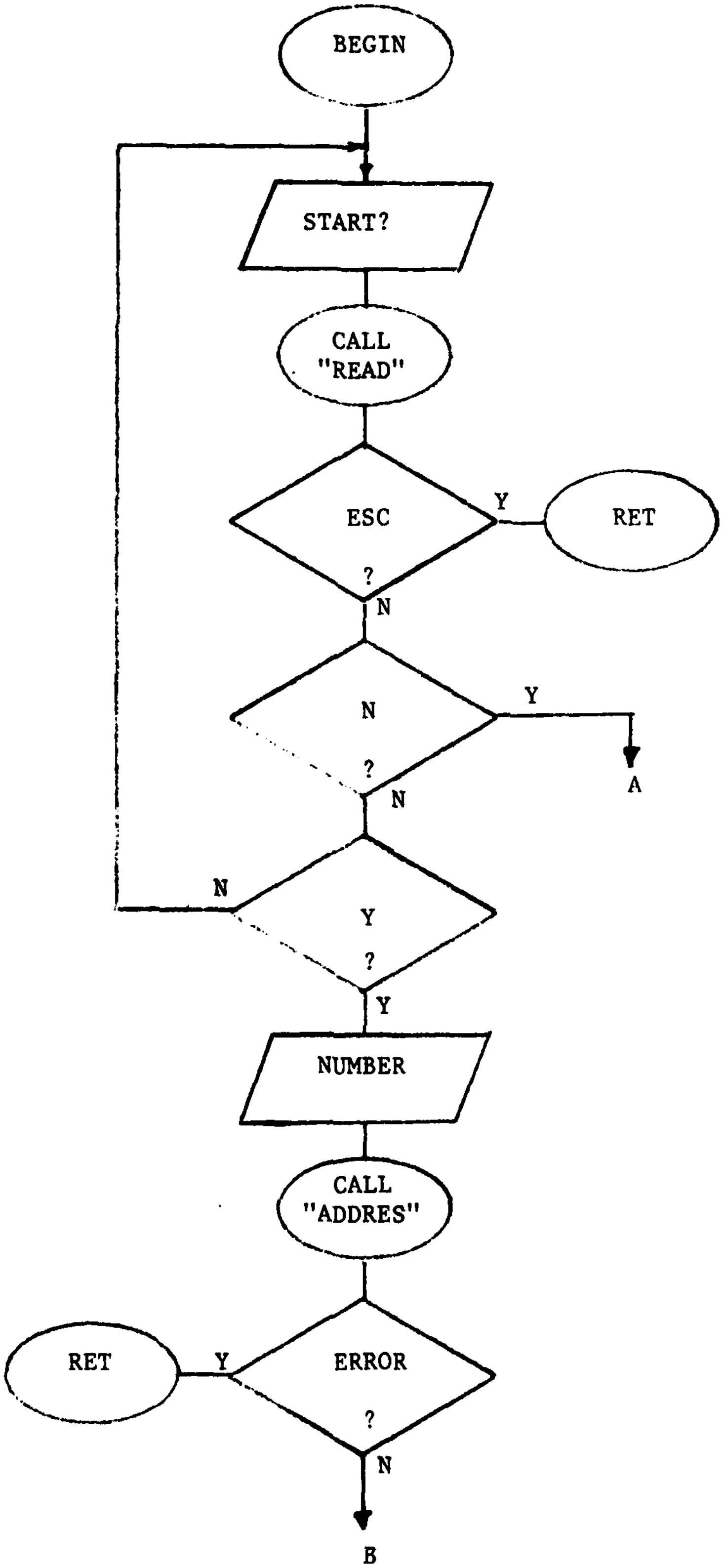


"TIME"

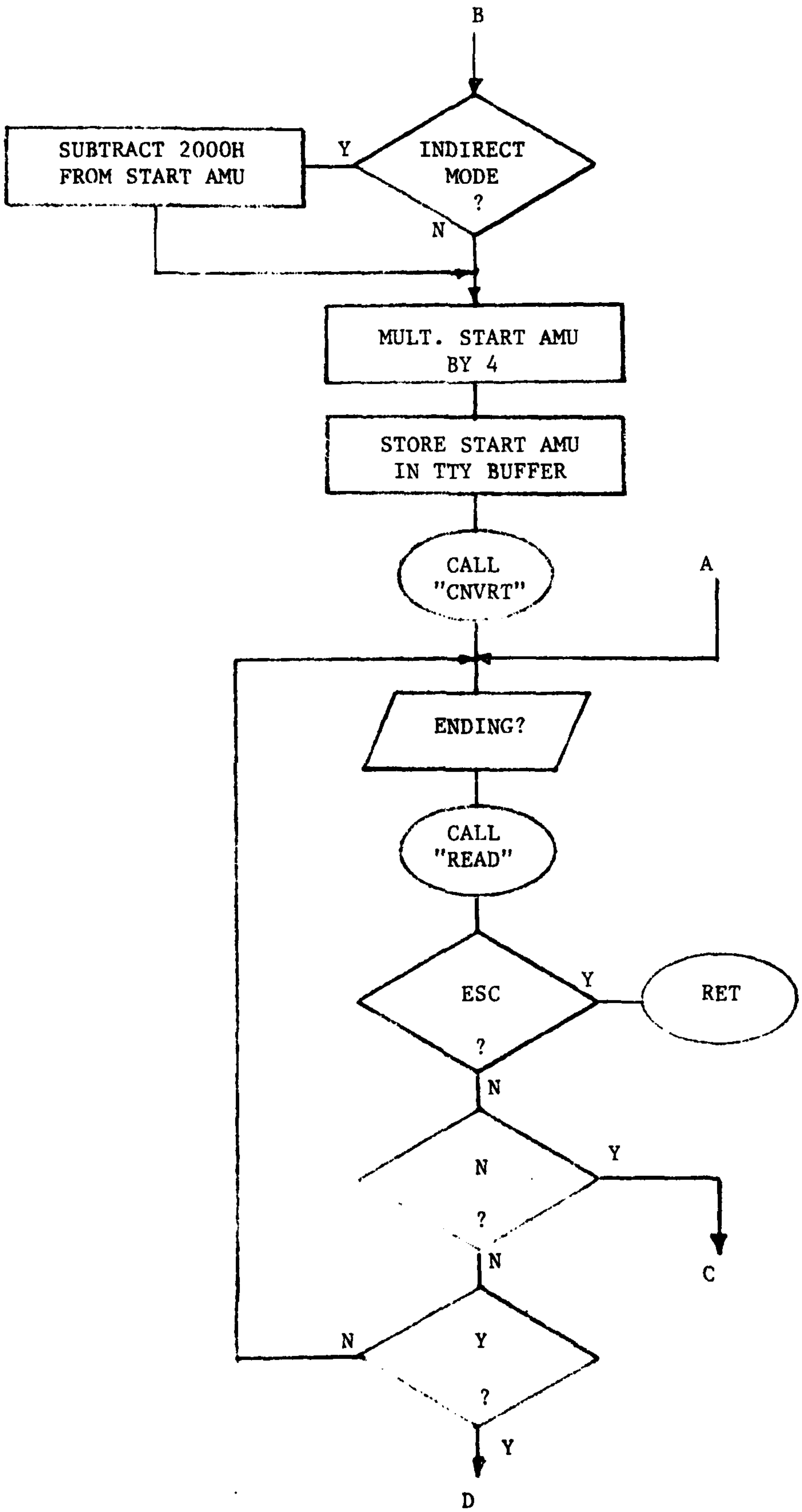
AD-A115 399 NORTHEASTERN UNIV BOSTON MASS ELECTRONICS RESEARCH LAB F/G 7/4
CONTROL ELECTRONICS FOR AIR-BORNE QUADRUPOLE ION MASS SPECTROME--ETC(U)
OCT 81 J S ROCHEFORT, R SUKYS F19628-78-C-0218
UNCLASSIFIED AFGL-TR-82-0056 NL

3 of 4
ADA

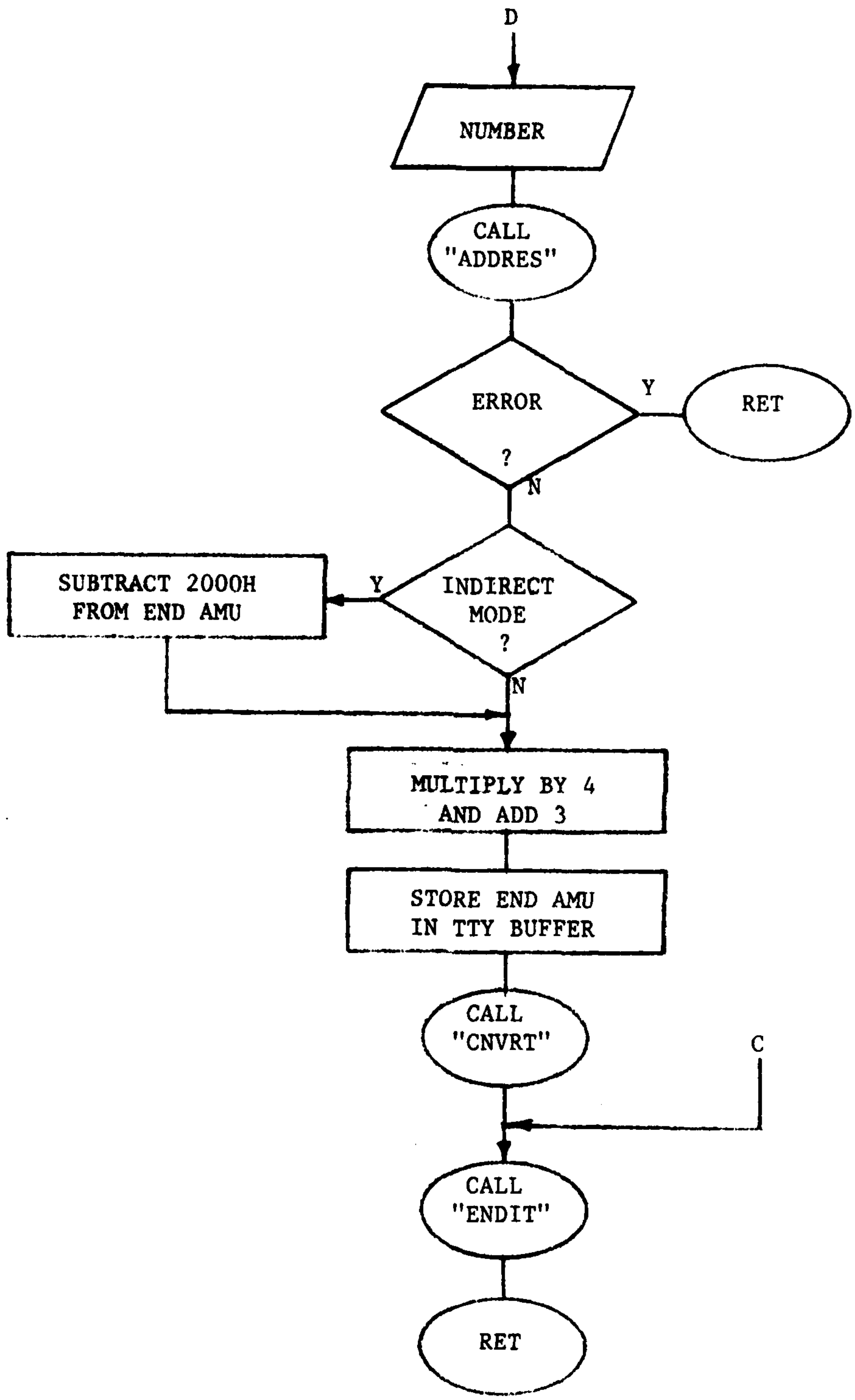

1.0
4.5
5.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6
MICROCOPY RESOLUTION TEST CHART


BEGIN
START?
CALL "READ"
ESC ?
Y
RET
N
N ?
Y
A
N
N
Y ?
Y
NUMBER
CALL "ADDRES"
RET
Y
ERROR ?
N
B

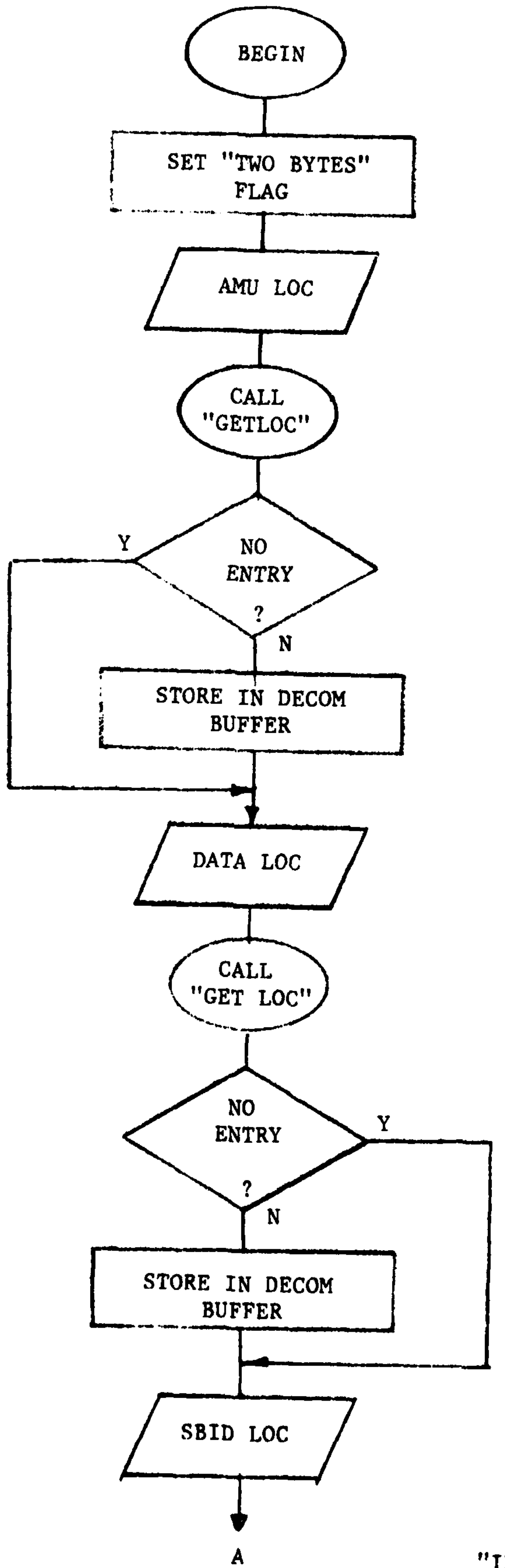
"AMU"


B
SUBTRACT 2000H FROM START AMU
Y
INDIRECT MODE ?
N
MULT. START AMU BY 4
STORE START AMU IN TTY BUFFER
CALL "CNVRT"
A
ENDING?
CALL "READ"
ESC ?
Y
RET
N
N ?
Y
C
N
N
Y ?
Y
D

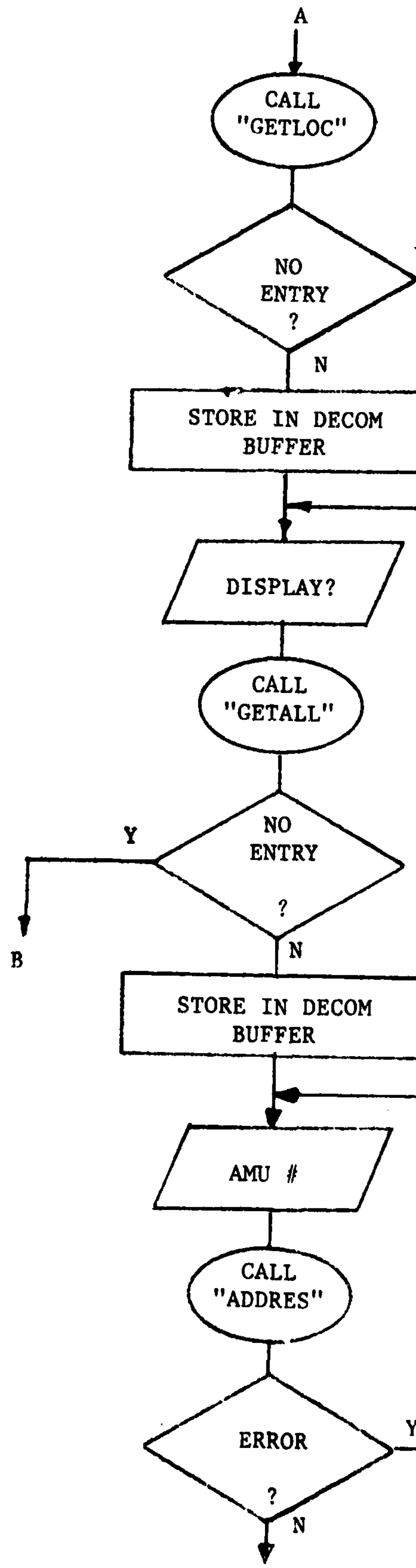
"AMU" CONT.


D
NUMBER
CALL "ADDRES"
ERROR ?
Y
RET
N
INDIRECT MODE ?
Y
SUBTRACT 2000H FROM END AMU
N
MULTIPLY BY 4 AND ADD 3
STORE END AMU IN TTY BUFFER
CALL "CNVRT"
C
CALL "ENDIT"
RET

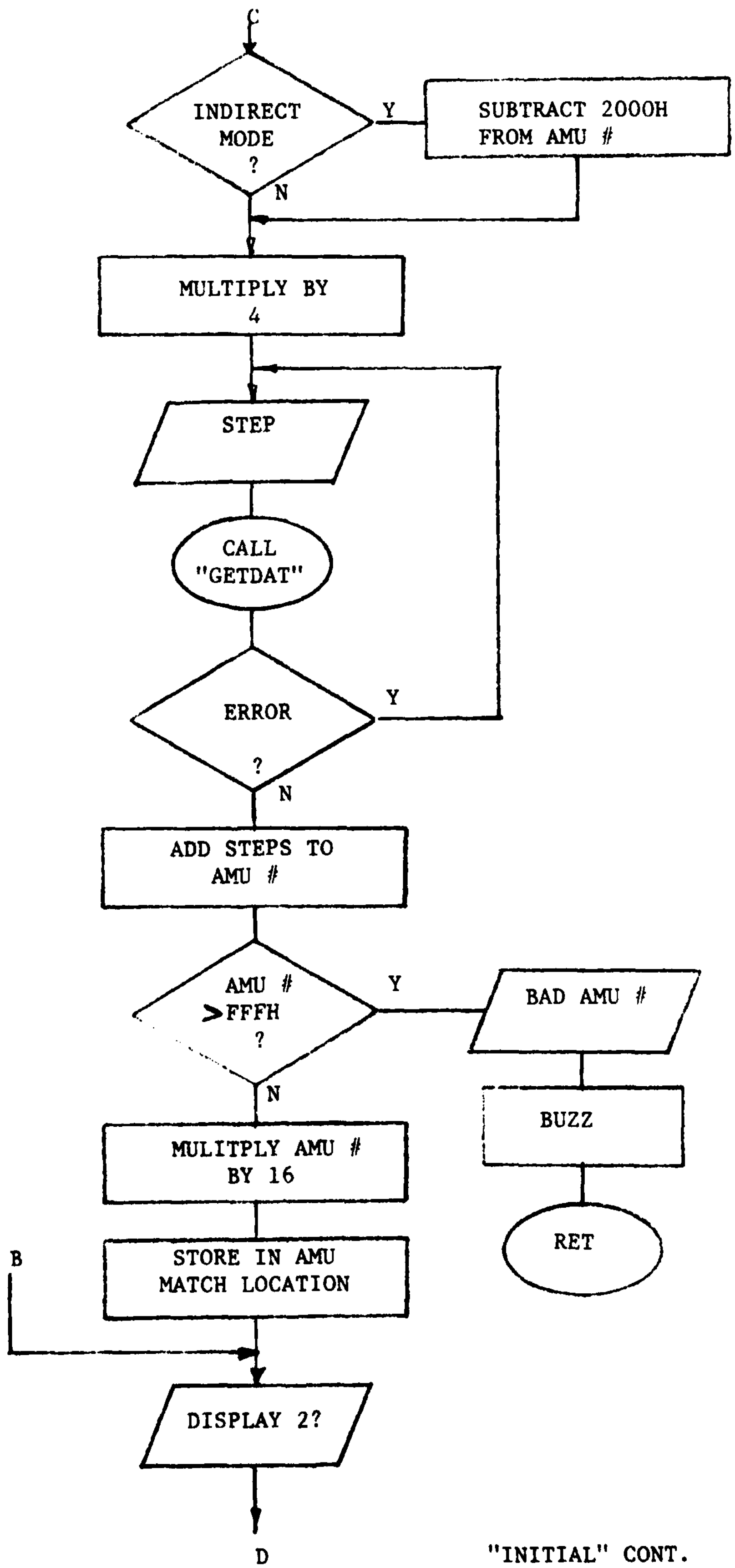
"AMU" CONT.


BEGIN
SET "TWO BYTES" FLAG
AMU LOC
CALL "GETLOC"
NO ENTRY ?
Y
N
STORE IN DECOM BUFFER
DATA LOC
CALL "GET LOC"
NO ENTRY ?
Y
N
STORE IN DECOM BUFFER
SBID LOC
A

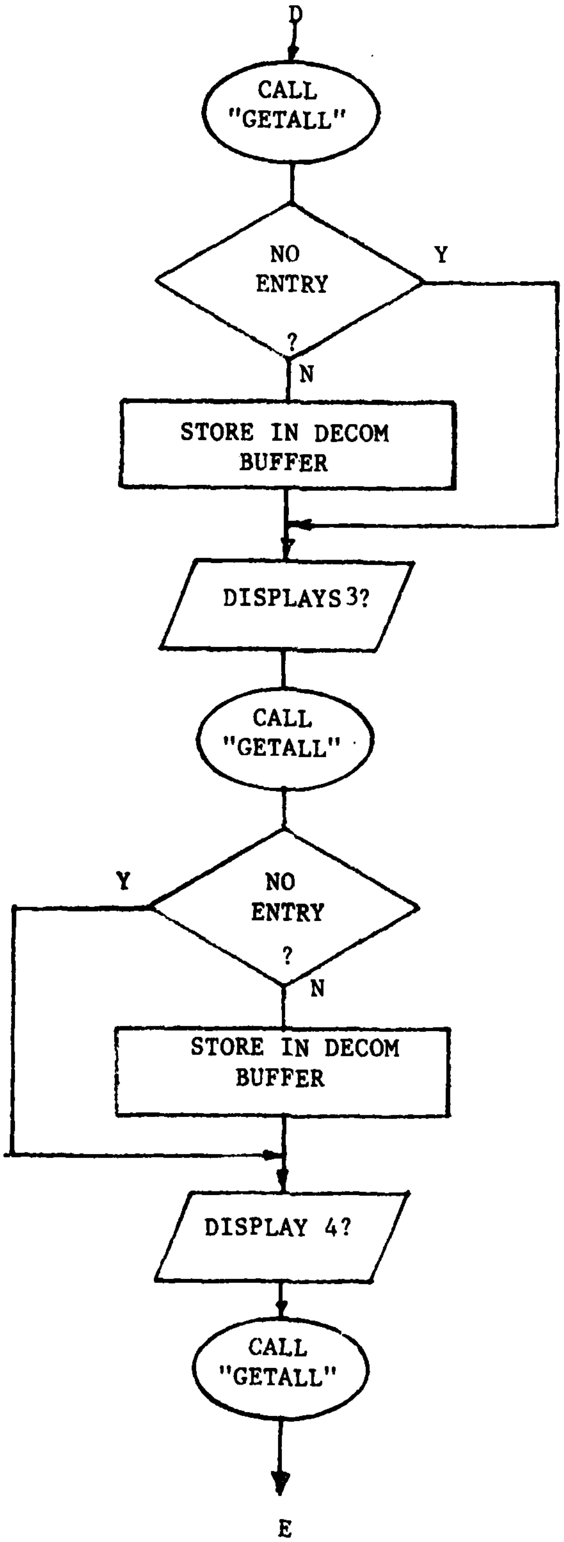
"INITIAL"



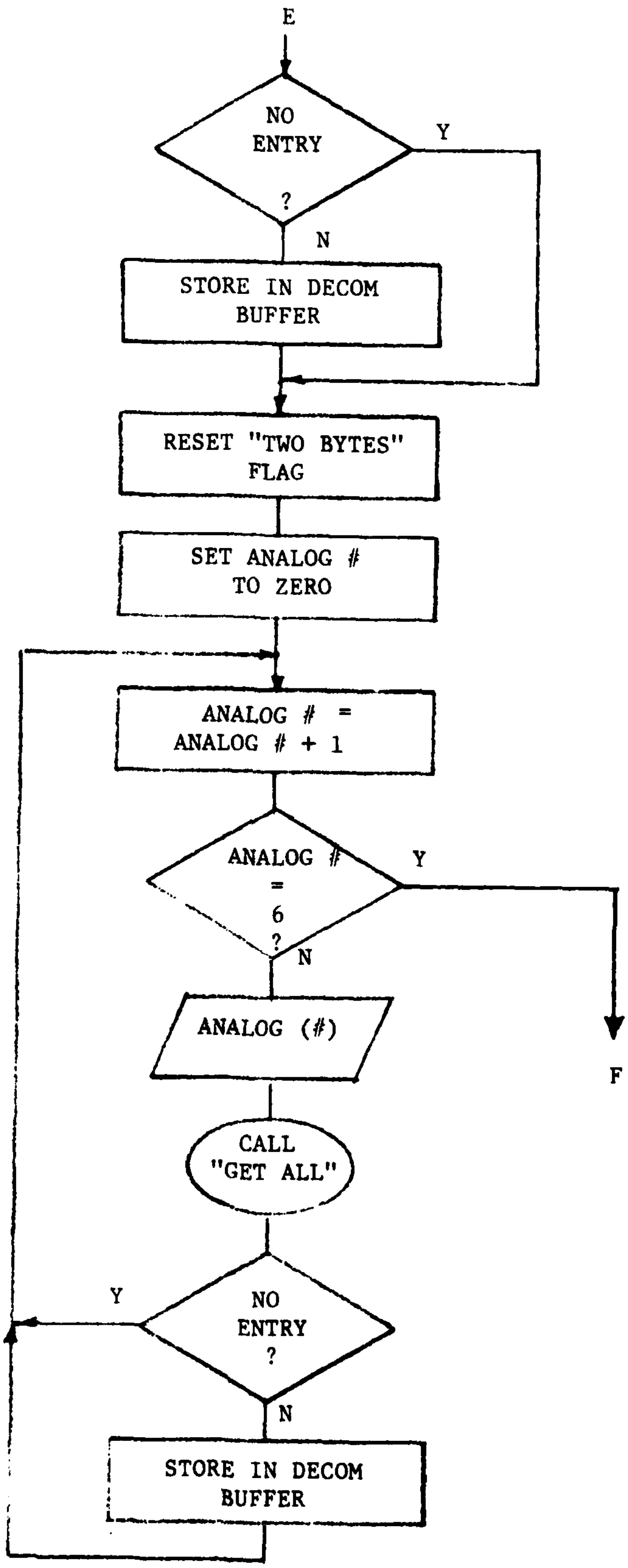
"INITIAL" CONT.



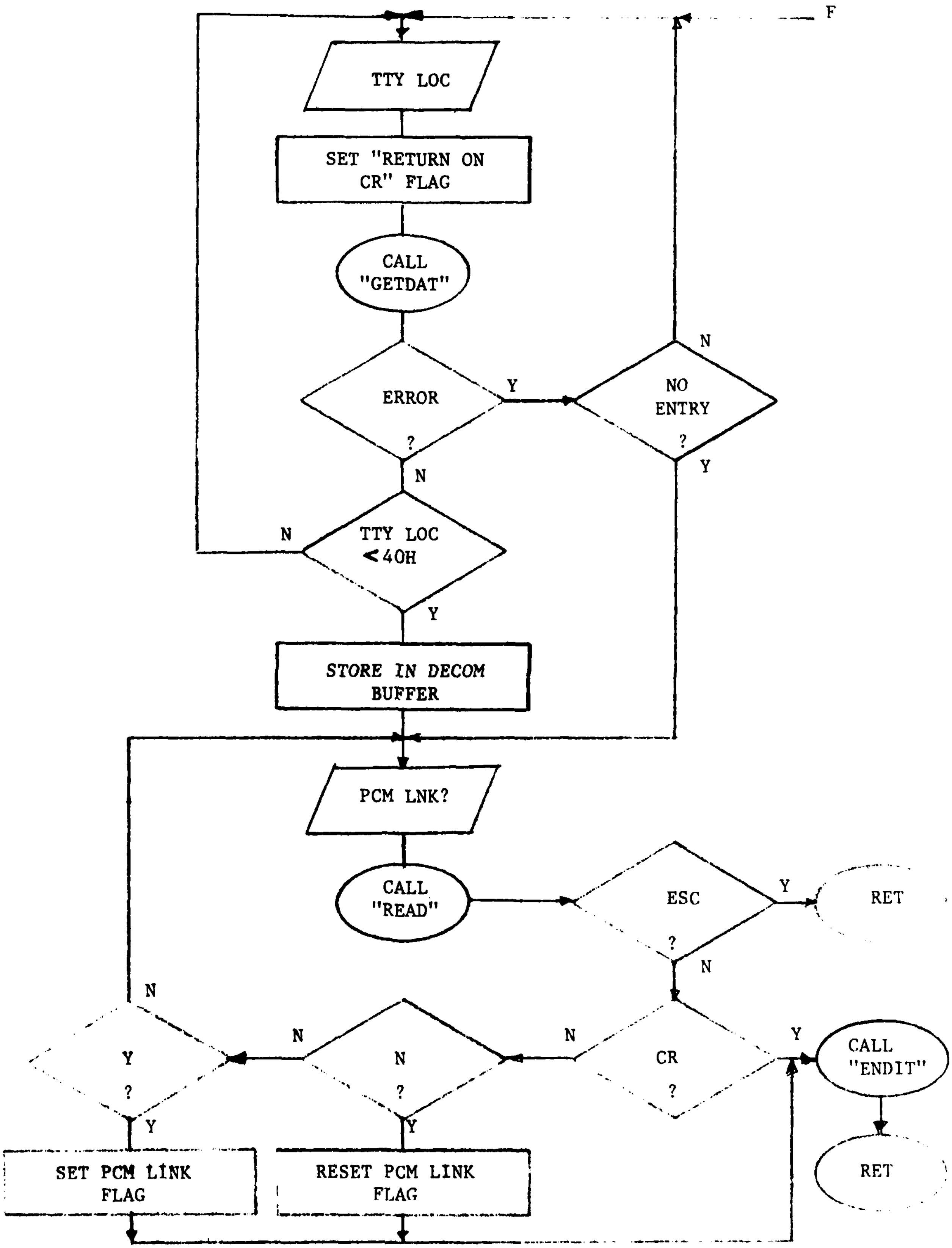
"INITIAL" CONT.



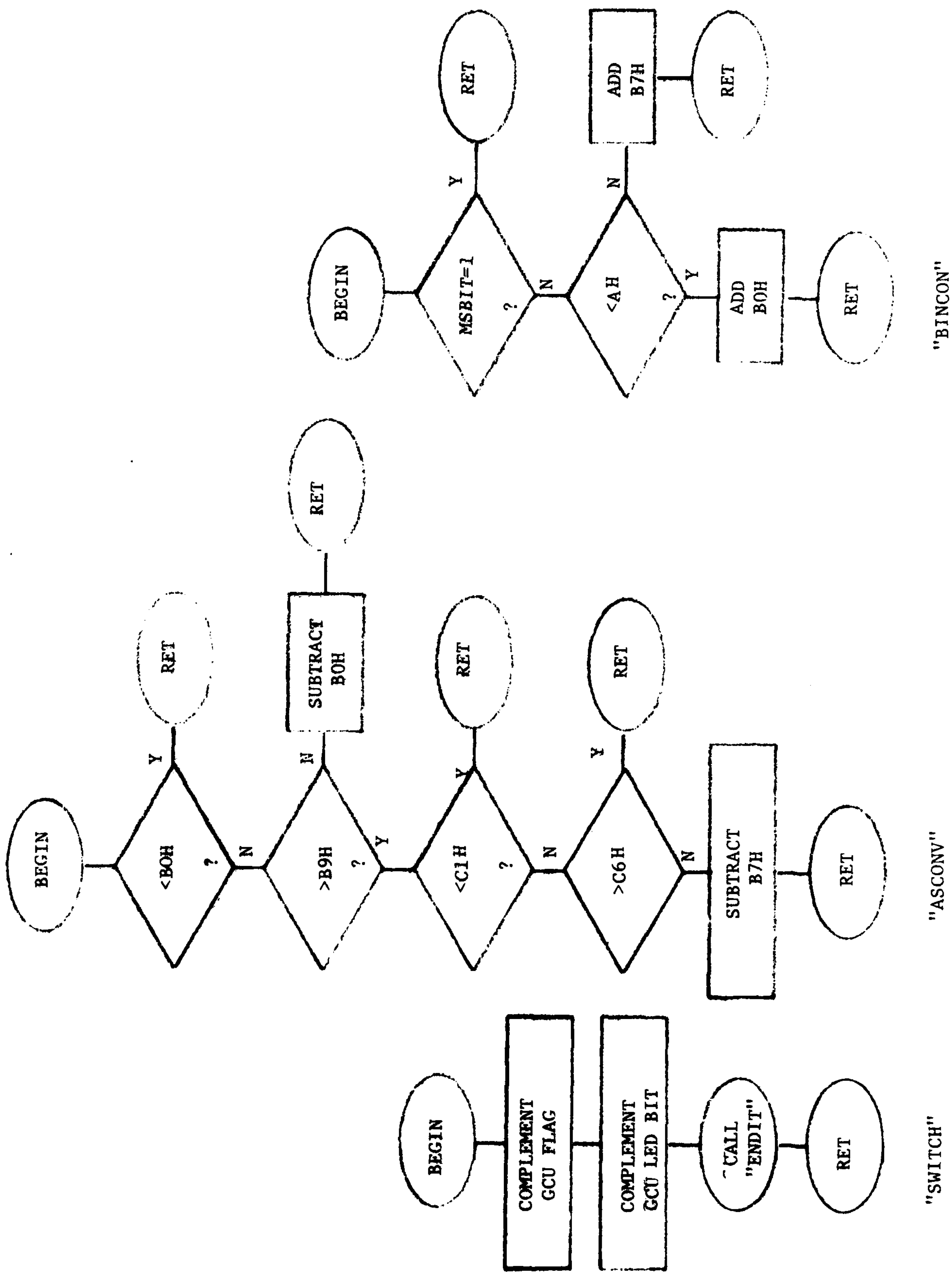
"INITIAL" CONT.

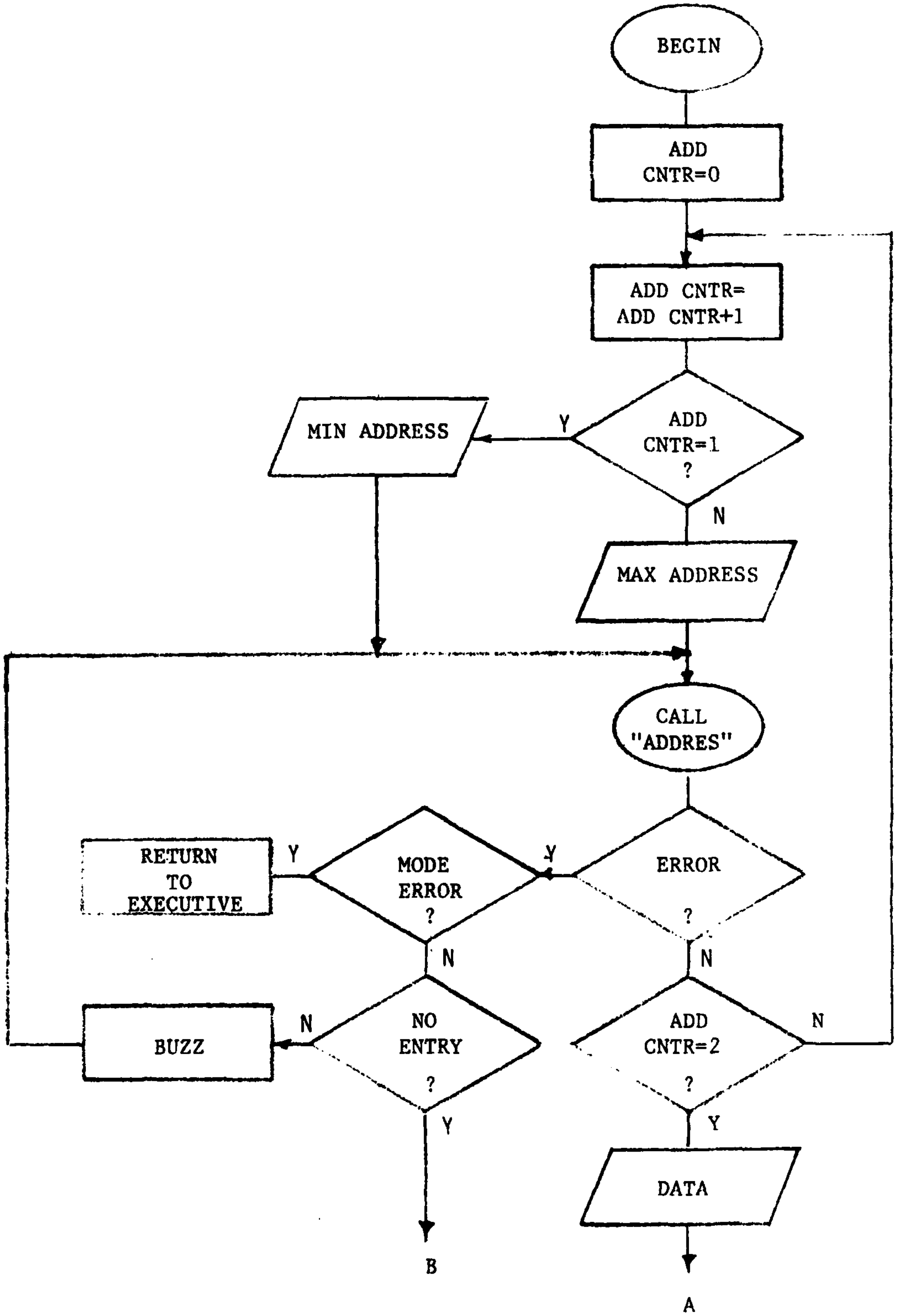


"INITIAL" CONT.



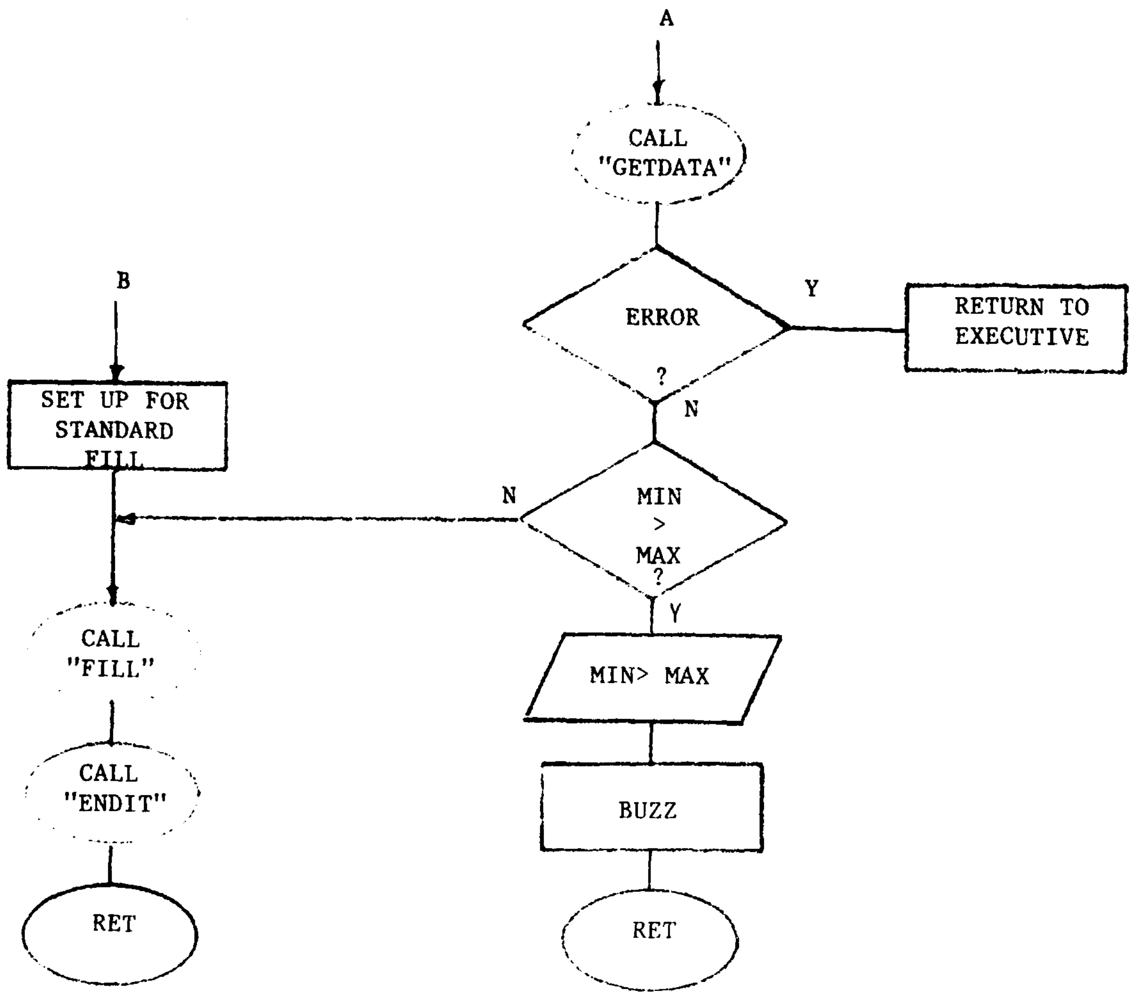
"INITIAL" CONT.


BEGIN
COMPLEMENT GCU FLAG
COMPLEMENT GCU LED BIT
CALL "ENDIT"
RET
"SWITCH"
BEGIN
<B0H ?
Y
RET
N
>B9H ?
N
SUBTRACT B0H
RET
Y
<C1H ?
Y
RET
N
>C6H
Y
RET
N
SUBTRACT B7H
RET
"ASCONV"
BEGIN
MSBIT=1 ?
Y
RET
N
<AH ?
N
ADD B7H
RET
Y
ADD B0H
RET
"BINCON"

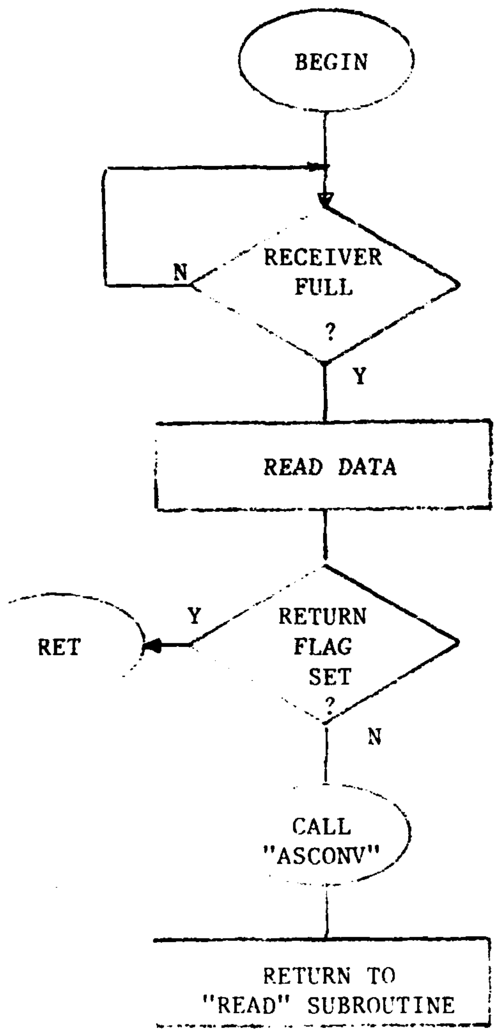


BEGIN
ADD CNTR=0
ADD CNTR=
ADD CNTR+1
ADD CNTR=1 ?
Y
N
MIN ADDRESS
MAX ADDRESS
CALL "ADDRES"
ERROR ?
Y
N
MODE ERROR ?
Y
N
RETURN TO EXECUTIVE
NO ENTRY ?
N
Y
BUZZ
ADD CNTR=2 ?
N
Y
DATA
B
A

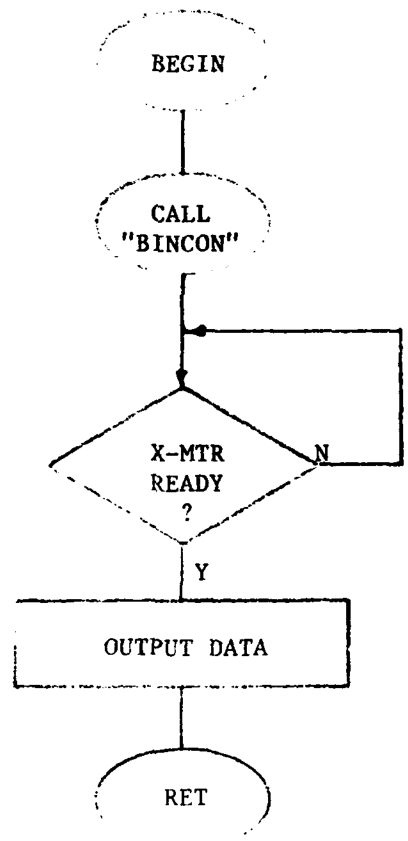
"FILLM"


A
CALL "GETDATA"
ERROR ?
Y
RETURN TO EXECUTIVE
N
MIN > MAX ?
N
Y
MIN> MAX
BUZZ
RET
B
SET UP FOR STANDARD FILL
CALL "FILL"
CALL "ENDIT"
RET

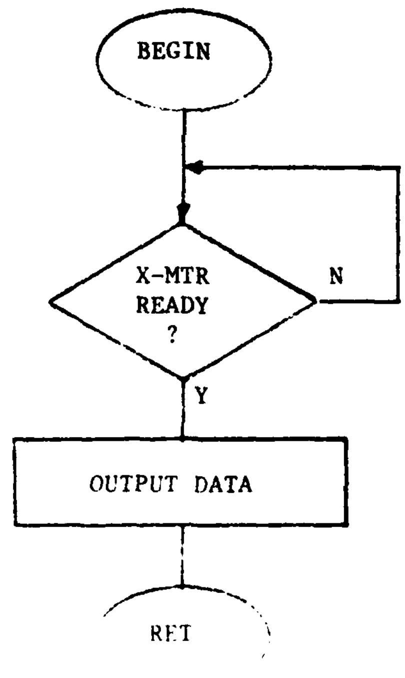
"FILLM" CONT.



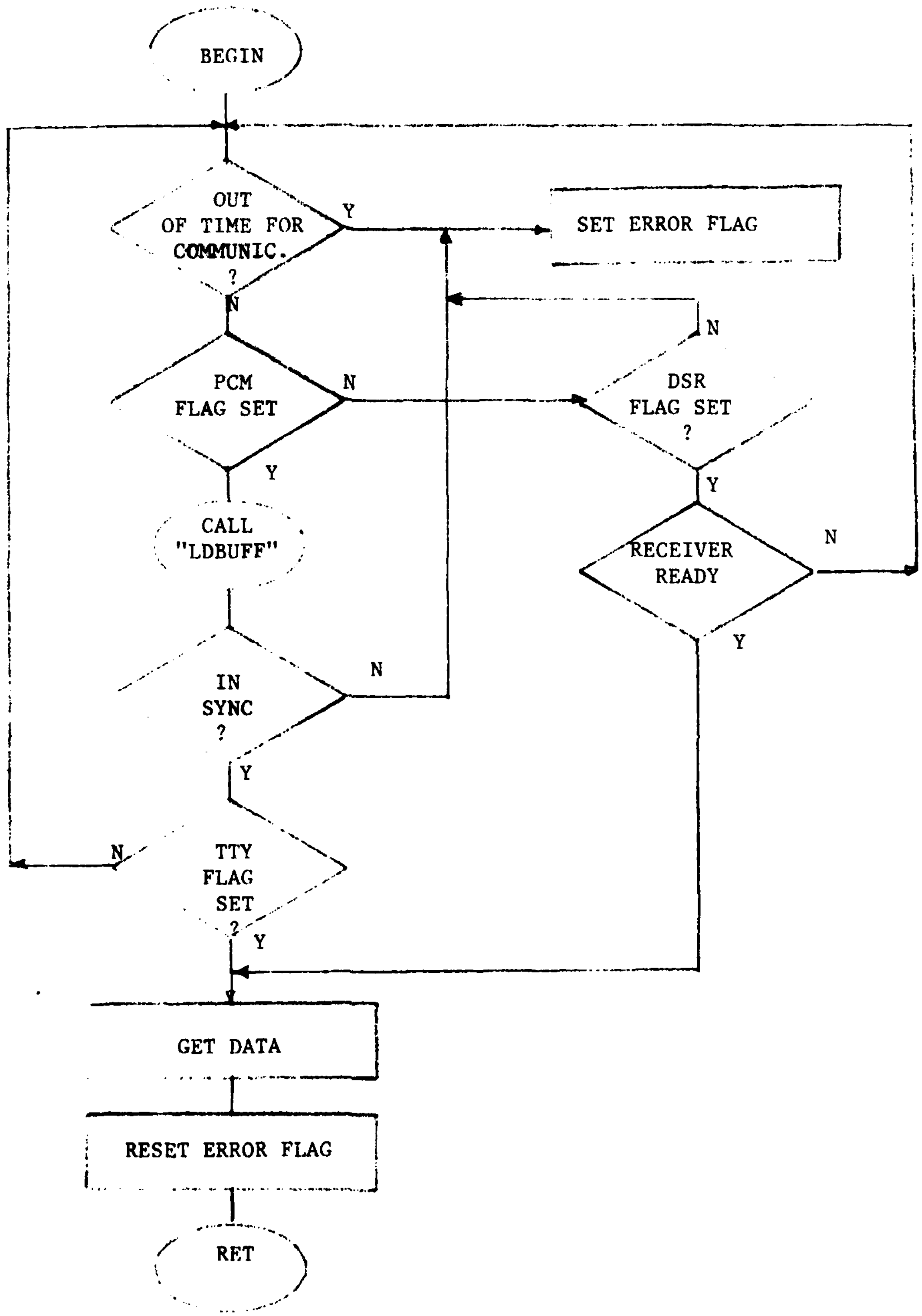
"TERMIN"



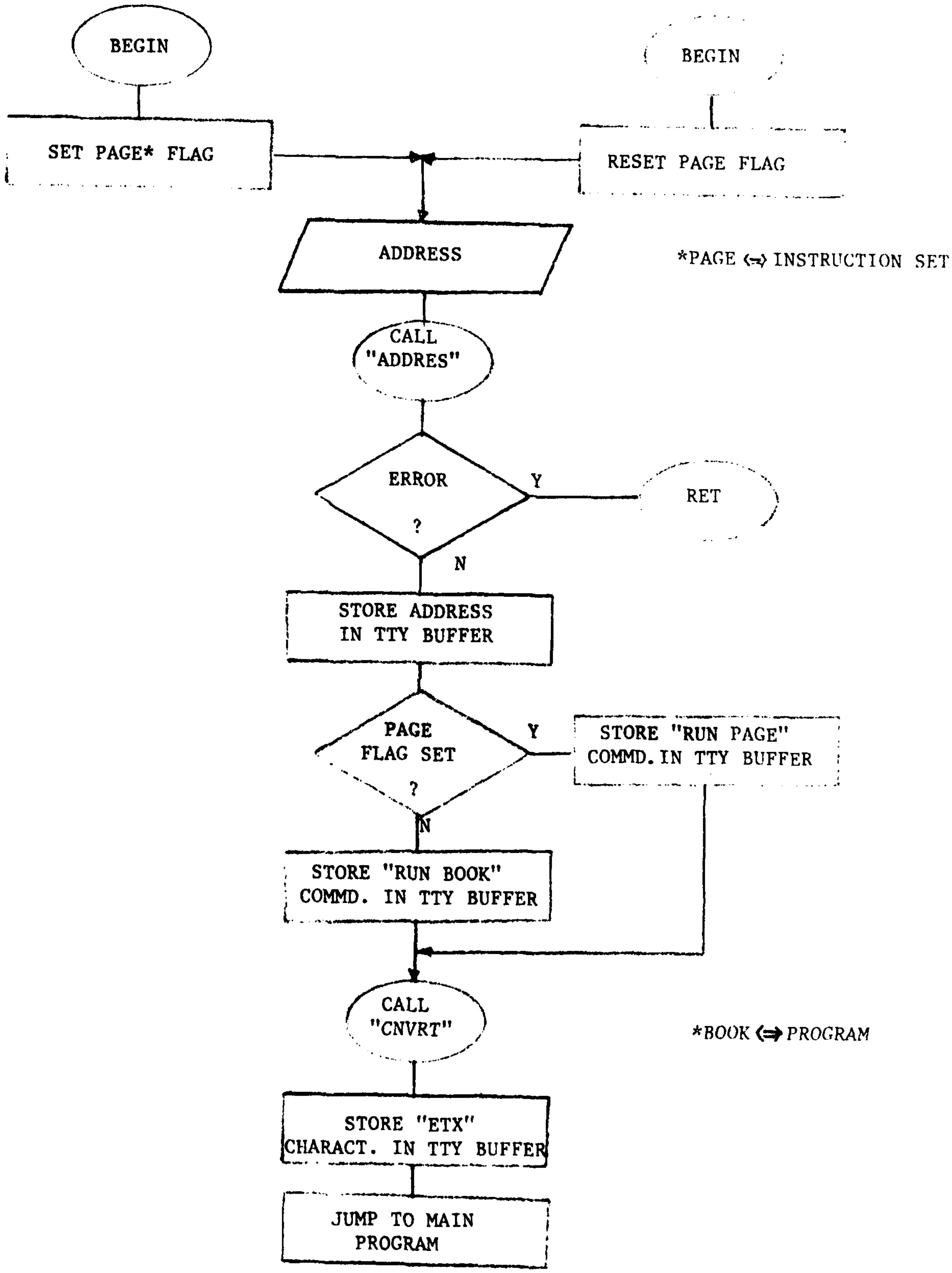
"TRMOUT"



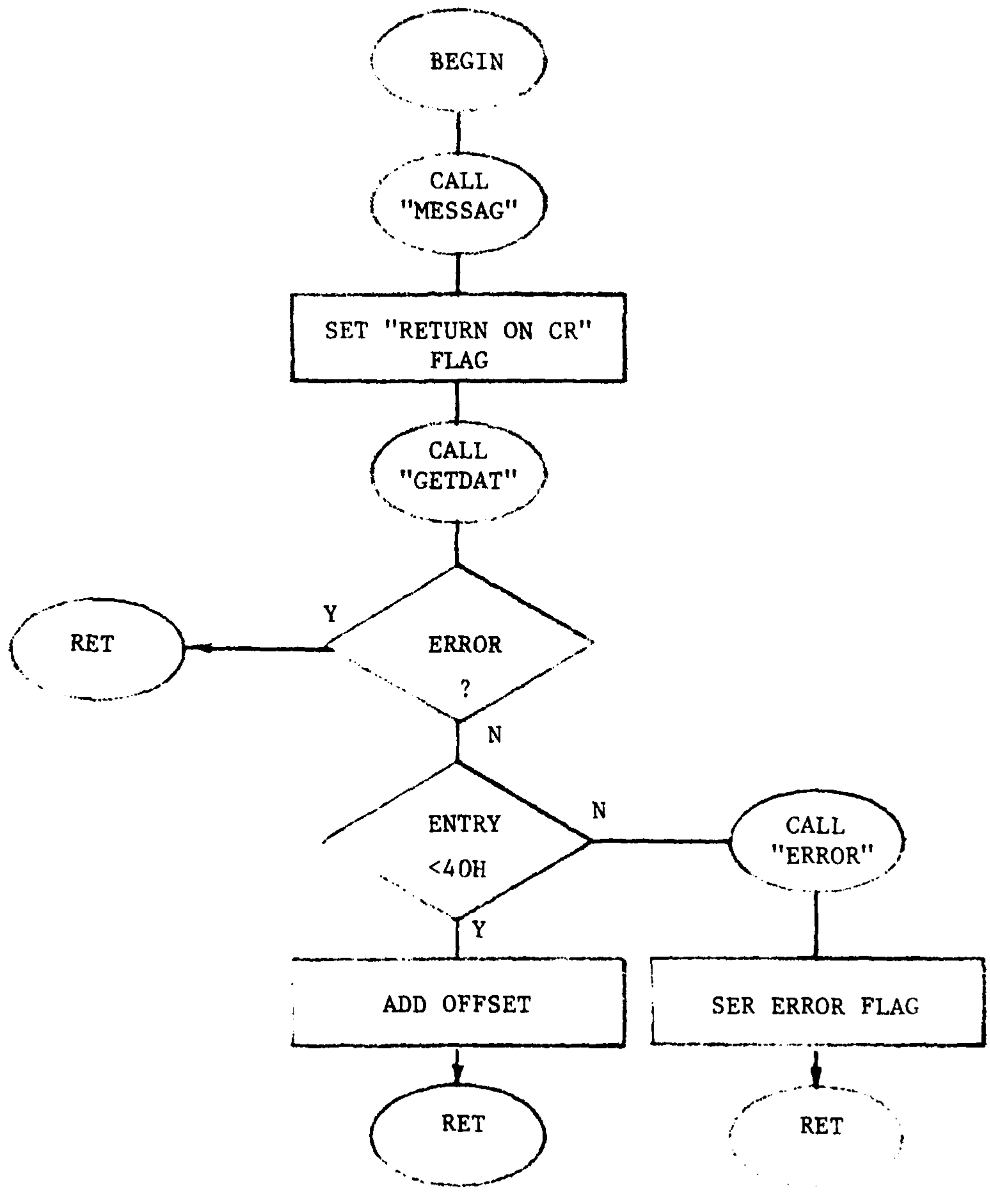
"TRNSMT"


BEGIN
OUT OF TIME FOR COMMUNIC. ?
Y
SET ERROR FLAG
N
PCM FLAG SET
N
DSR FLAG SET ?
N
Y
Y
CALL "LDBUFF"
RECEIVER READY
N
IN SYNC ?
N
Y
Y
TTY FLAG SET ?
N
Y
GET DATA
RESET ERROR FLAG
RET

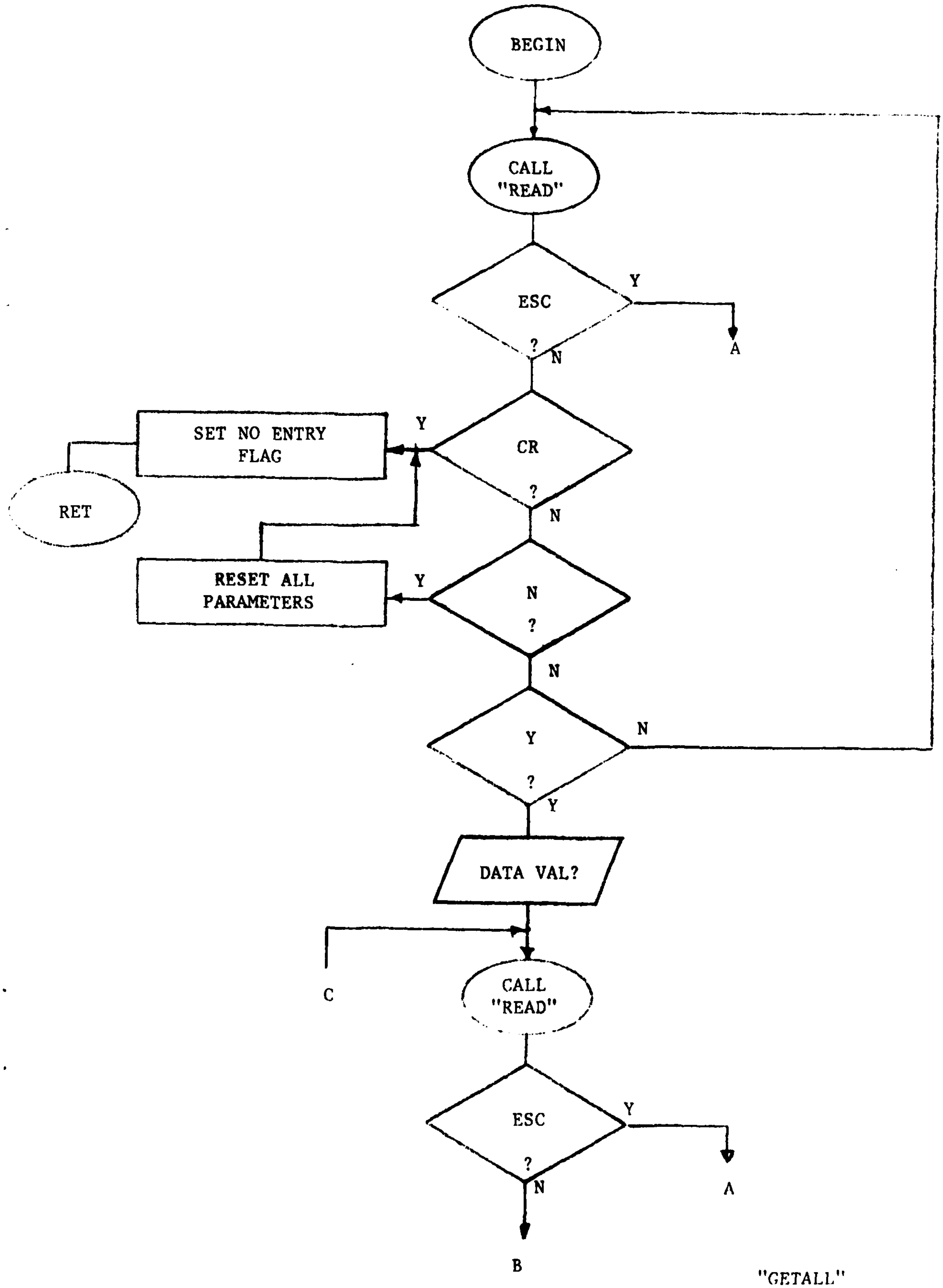
"RECEV"


BEGIN
BEGIN
SET PAGE* FLAG
RESET PAGE FLAG
ADDRESS
*PAGE ⟺ INSTRUCTION SET
CALL "ADDRES"
ERROR ?
Y
RET
N
STORE ADDRESS IN TTY BUFFER
PAGE FLAG SET ?
Y
STORE "RUN PAGE" COMMD. IN TTY BUFFER
N
STORE "RUN BOOK" COMMD. IN TTY BUFFER
CALL "CNVRT"
*BOOK ⟺ PROGRAM
STORE "ETX" CHARACT. IN TTY BUFFER
JUMP TO MAIN PROGRAM

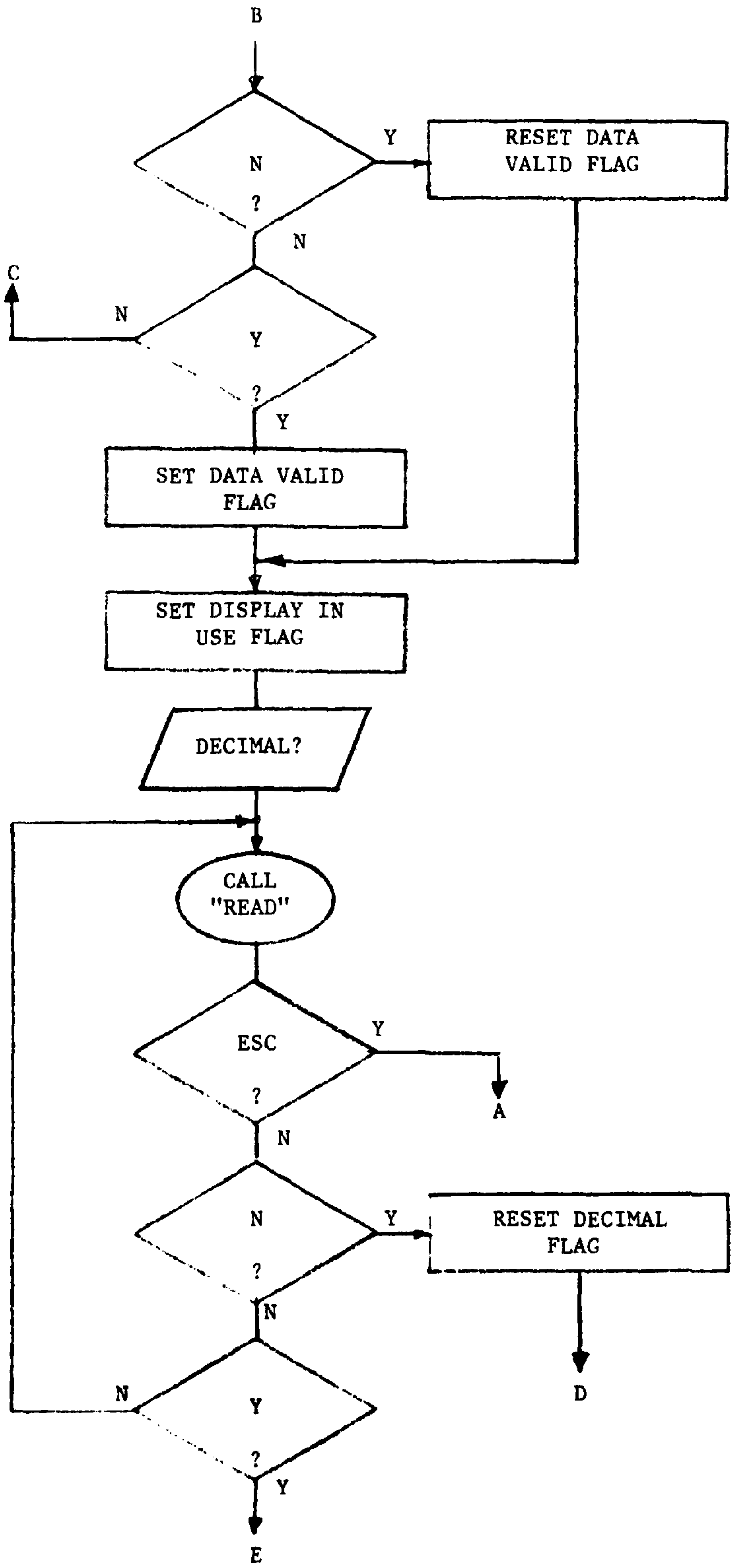
"RPAGE", "RBOOK"


BEGIN
CALL "MESSAG"
SET "RETURN ON CR" FLAG
CALL "GETDAT"
ERROR ?
Y
RET
N
ENTRY <40H
N
CALL "ERROR"
Y
ADD OFFSET
SER ERROR FLAG
RET
RET

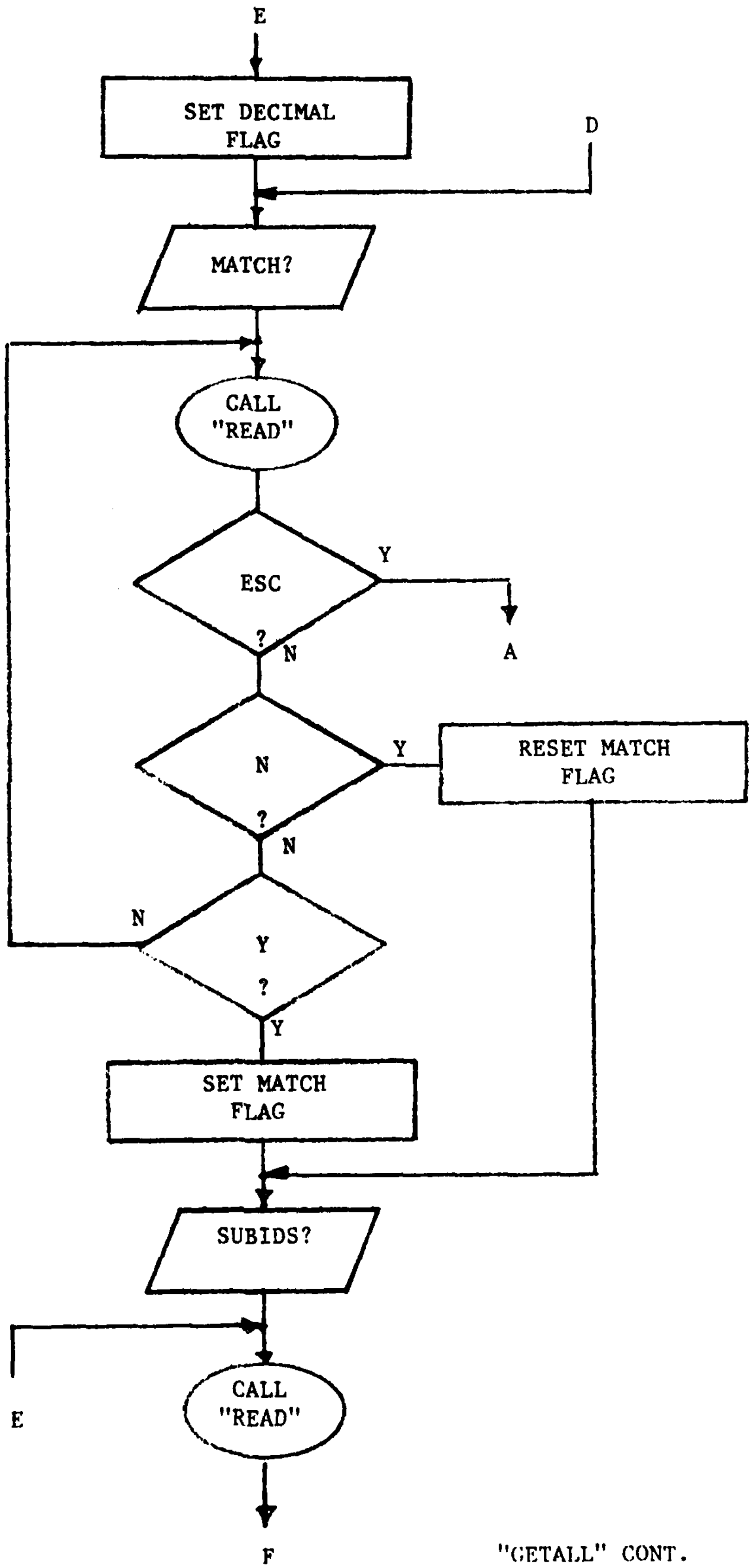
"GETLOC"


BEGIN
CALL "READ"
ESC ?
Y
A
N
CR ?
Y
SET NO ENTRY FLAG
RET
N
N ?
Y
RESET ALL PARAMETERS
N
Y ?
N
Y
DATA VAL?
C
CALL "READ"
ESC ?
Y
A
N
B

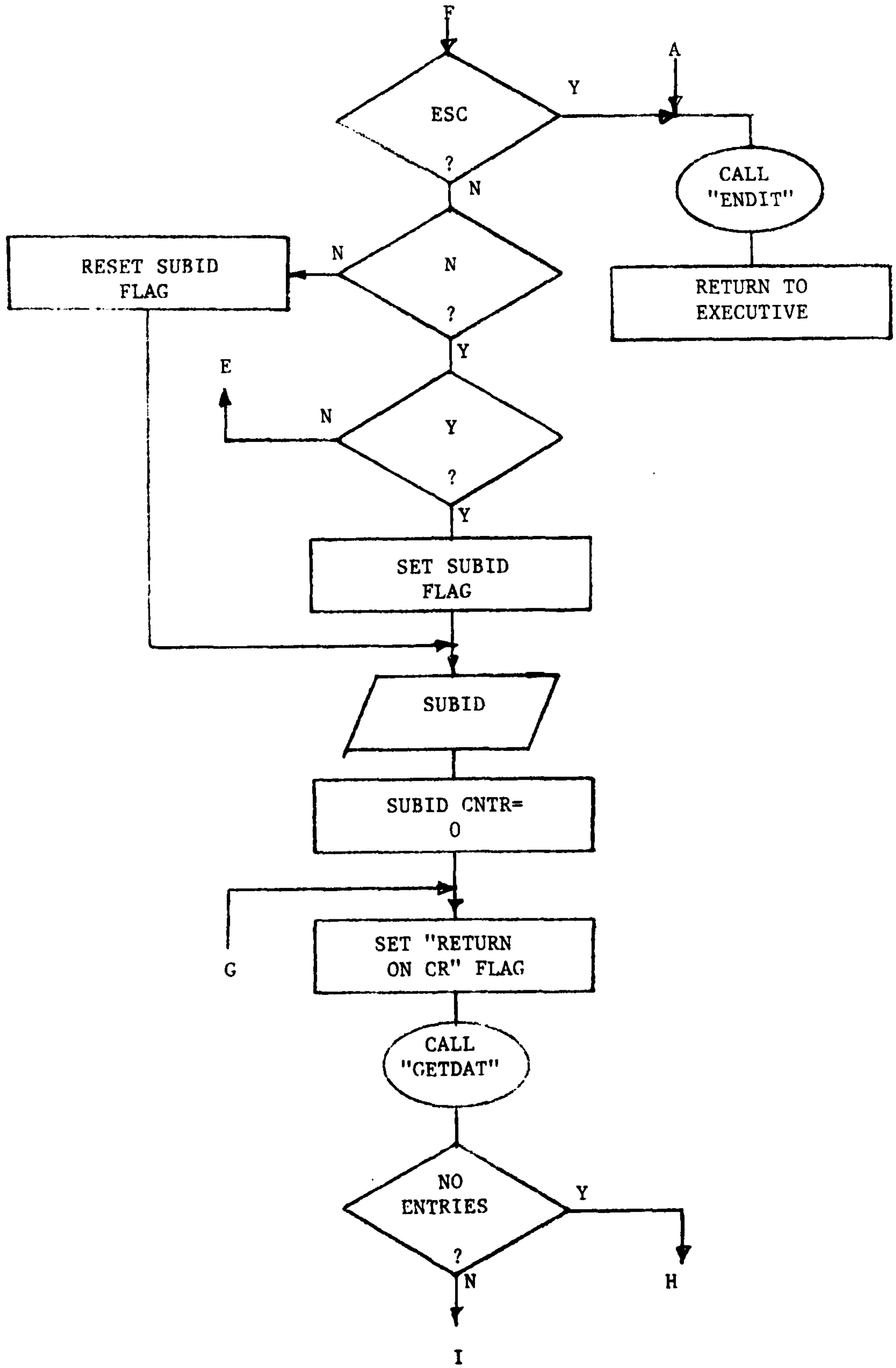
"GETALL"


B
N ?
Y
RESET DATA VALID FLAG
N
C
N
Y ?
Y
SET DATA VALID FLAG
SET DISPLAY IN USE FLAG
DECIMAL?
CALL "READ"
ESC ?
Y
A
N
N ?
Y
RESET DECIMAL FLAG
N
D
N
Y ?
Y
E

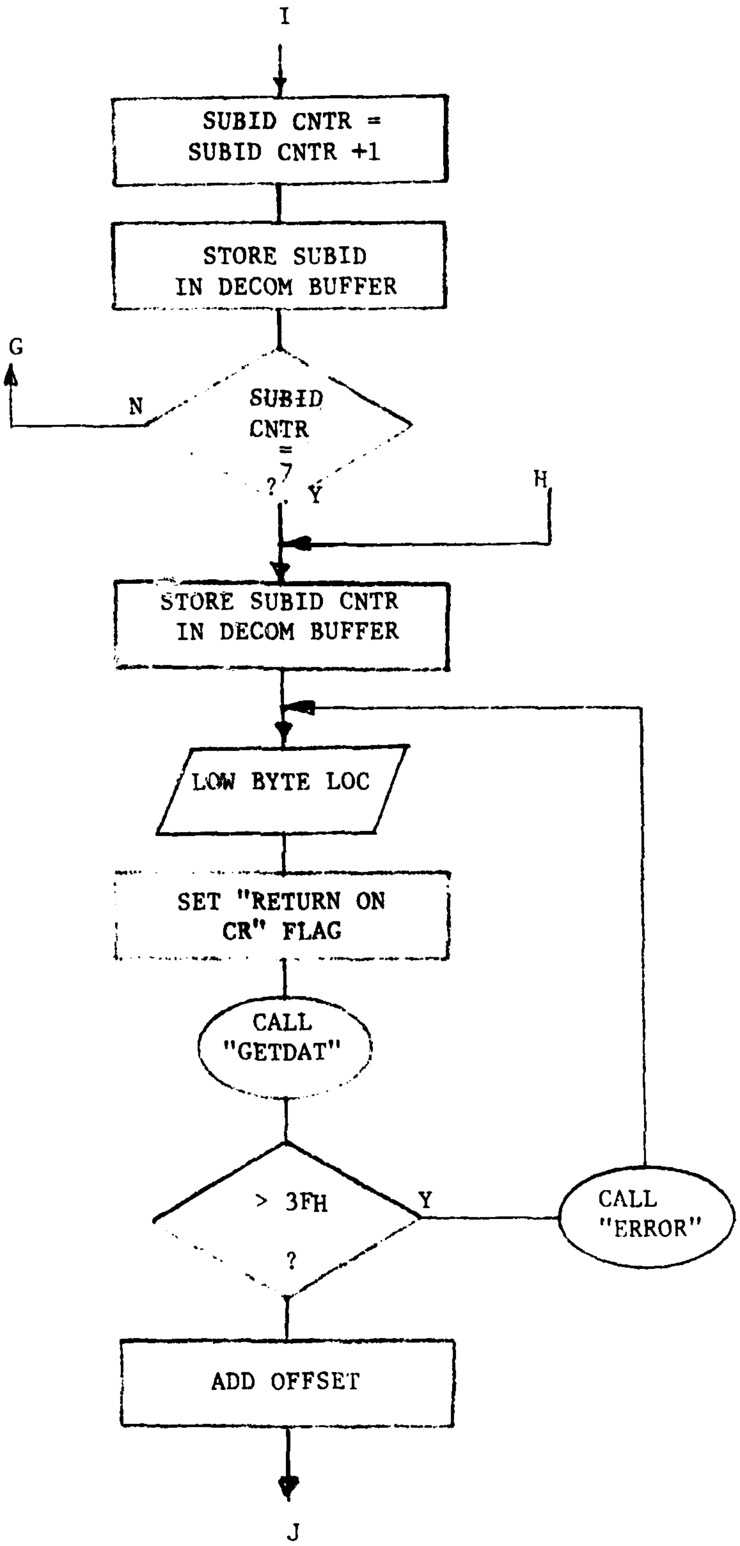
"GETALL" CONT.


E
SET DECIMAL FLAG
D
MATCH?
CALL "READ"
ESC ?
Y
A
N
N ?
Y
RESET MATCH FLAG
N
N
Y ?
Y
SET MATCH FLAG
SUBIDS?
CALL "READ"
E
F

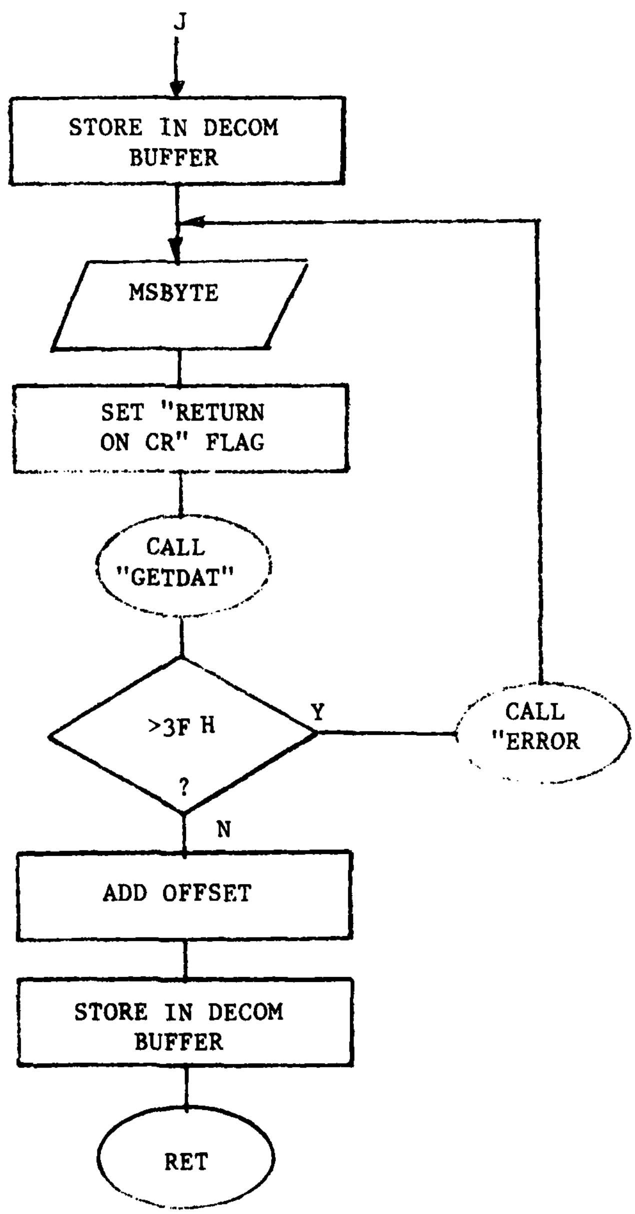
"GETALL" CONT.


F
ESC ?
Y
A
CALL "ENDIT"
RETURN TO EXECUTIVE
N
N ?
N
RESET SUBID FLAG
Y
E
N
Y ?
Y
SET SUBID FLAG
SUBID
SUBID CNTR= 0
G
SET "RETURN ON CR" FLAG
CALL "GETDAT"
NO ENTRIES ?
Y
H
N
I

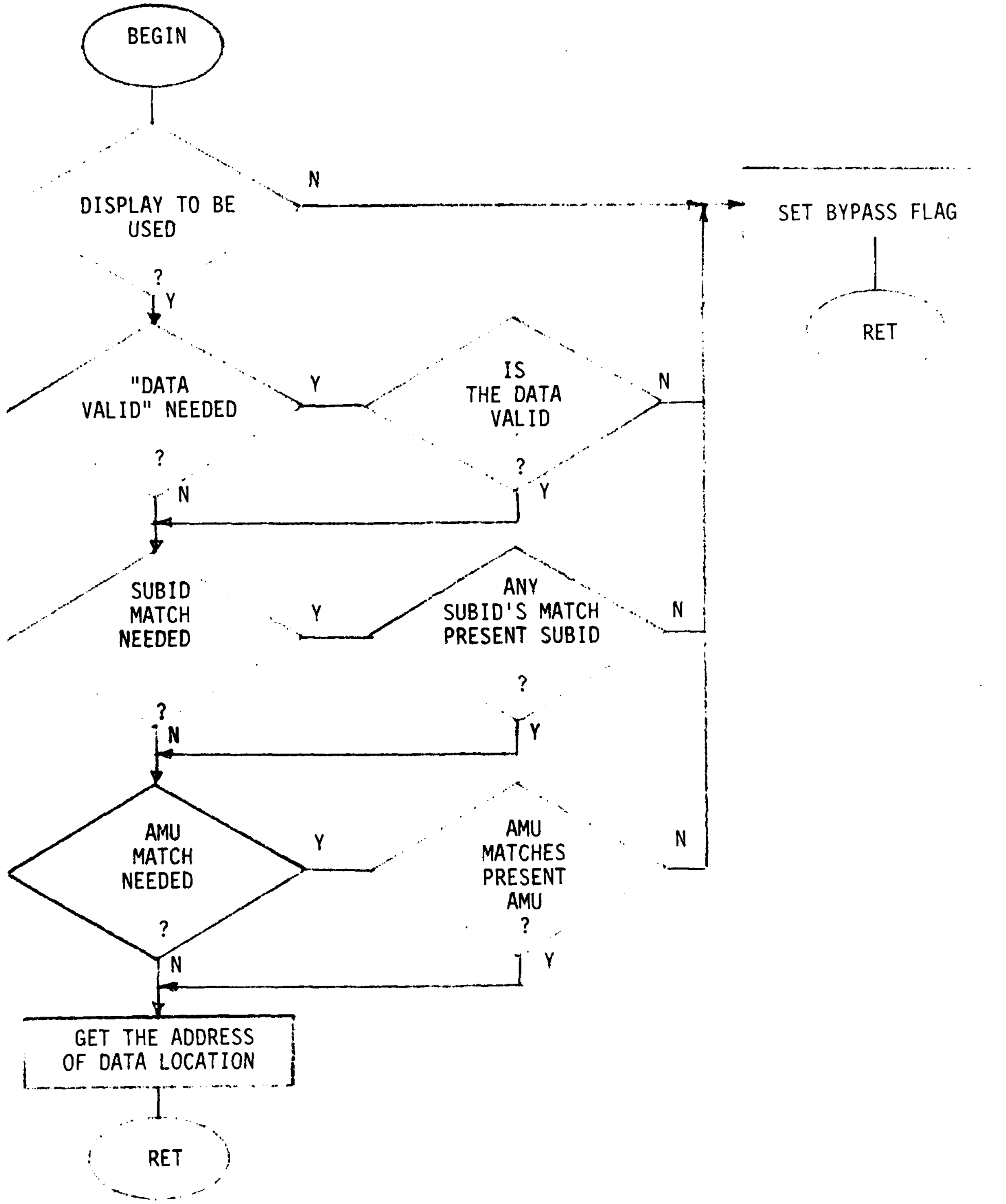
"GETALL" CONT.


I
SUBID CNTR = SUBID CNTR +1
STORE SUBID IN DECOM BUFFER
G
N
SUBID CNTR = 7 ?
Y
H
STORE SUBID CNTR IN DECOM BUFFER
LOW BYTE LOC
SET "RETURN ON CR" FLAG
CALL "GETDAT"
> 3FH ?
Y
CALL "ERROR"
ADD OFFSET
J

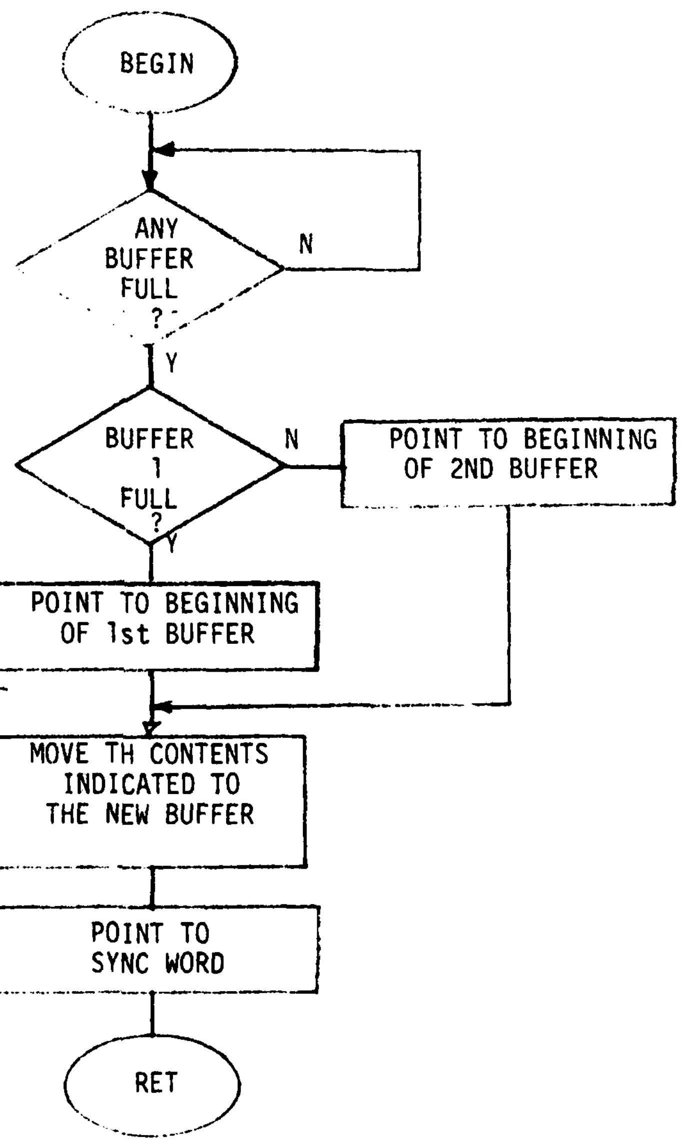
"GETALL" CONT.


J
STORE IN DECOM BUFFER
MSBYTE
SET "RETURN ON CR" FLAG
CALL "GETDAT"
>3F H ?
Y
CALL "ERROR
N
ADD OFFSET
STORE IN DECOM BUFFER
RET

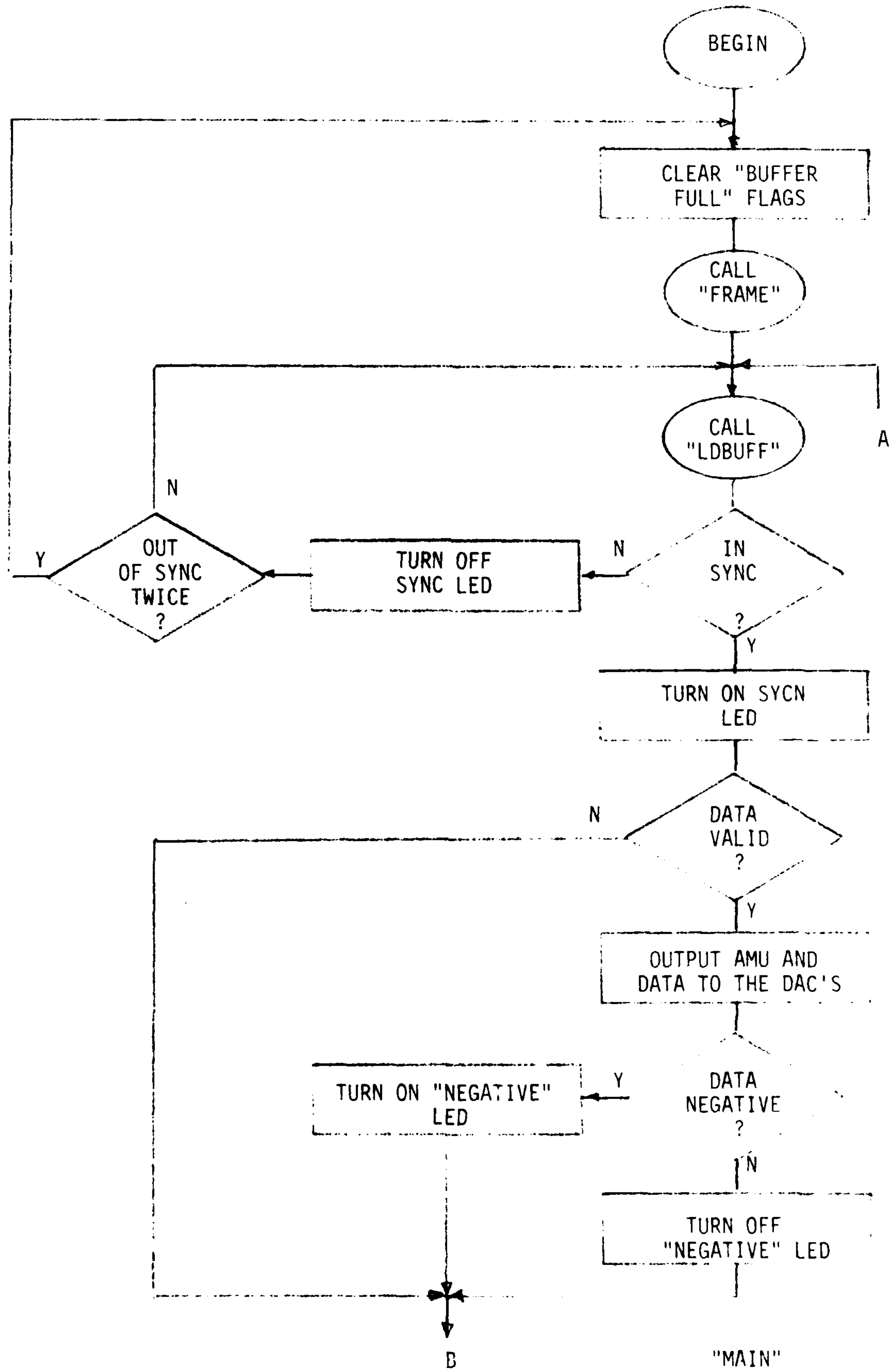
"GETALL" CONT.

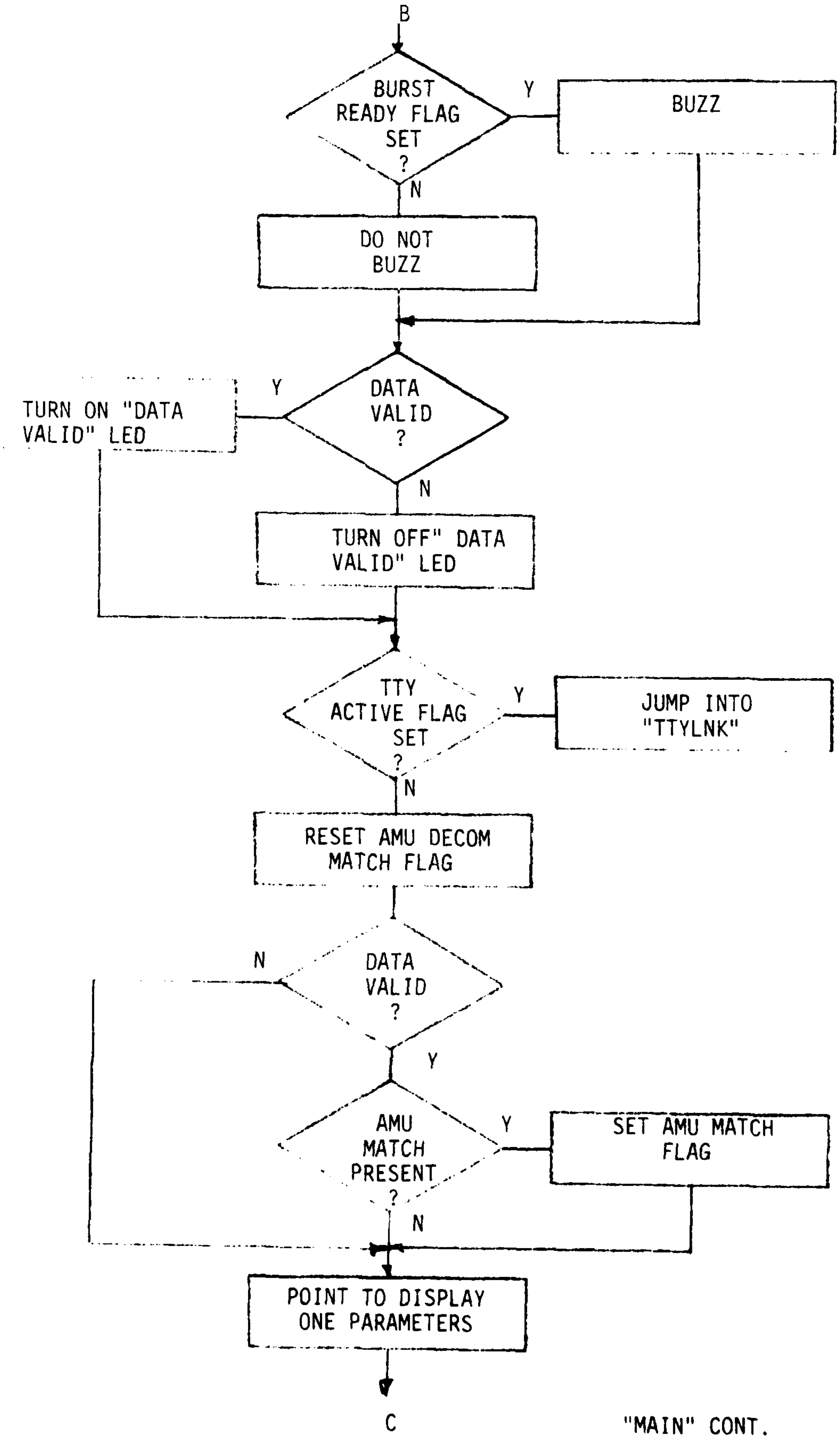

BEGIN
DISPLAY TO BE USED ?
N
Y
SET BYPASS FLAG
RET
"DATA VALID" NEEDED ?
Y
IS THE DATA VALID ?
N
N
Y
SUBID MATCH NEEDED ?
Y
ANY SUBID'S MATCH PRESENT SUBID ?
N
N
Y
AMU MATCH NEEDED ?
Y
AMU MATCHES PRESENT AMU ?
N
N
Y
GET THE ADDRESS OF DATA LOCATION
RET

"ALLDEF"

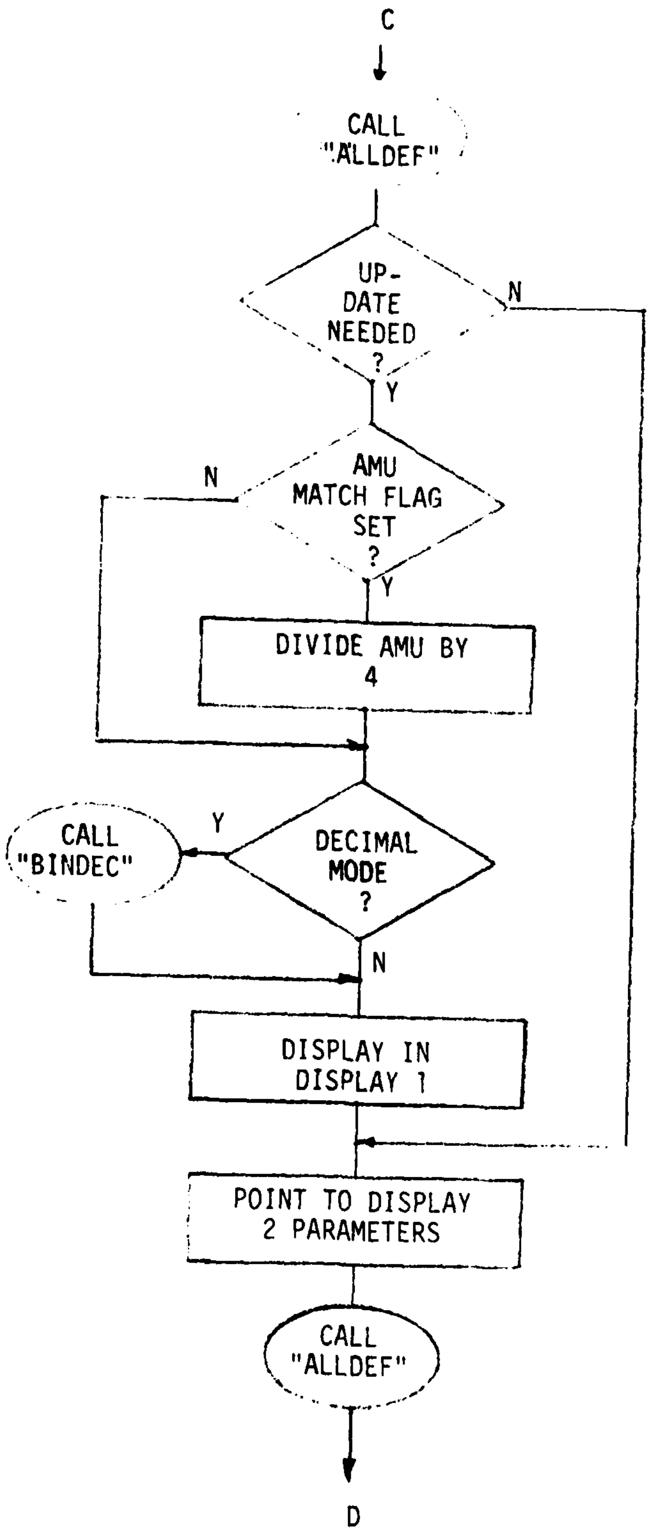

BEGIN
ANY BUFFER FULL ?
N
Y
BUFFER 1 FULL ?
N
POINT TO BEGINNING OF 2ND BUFFER
Y
POINT TO BEGINNING OF 1st BUFFER
MOVE TH CONTENTS INDICATED TO THE NEW BUFFER
POINT TO SYNC WORD
RET

"LDBUFF"

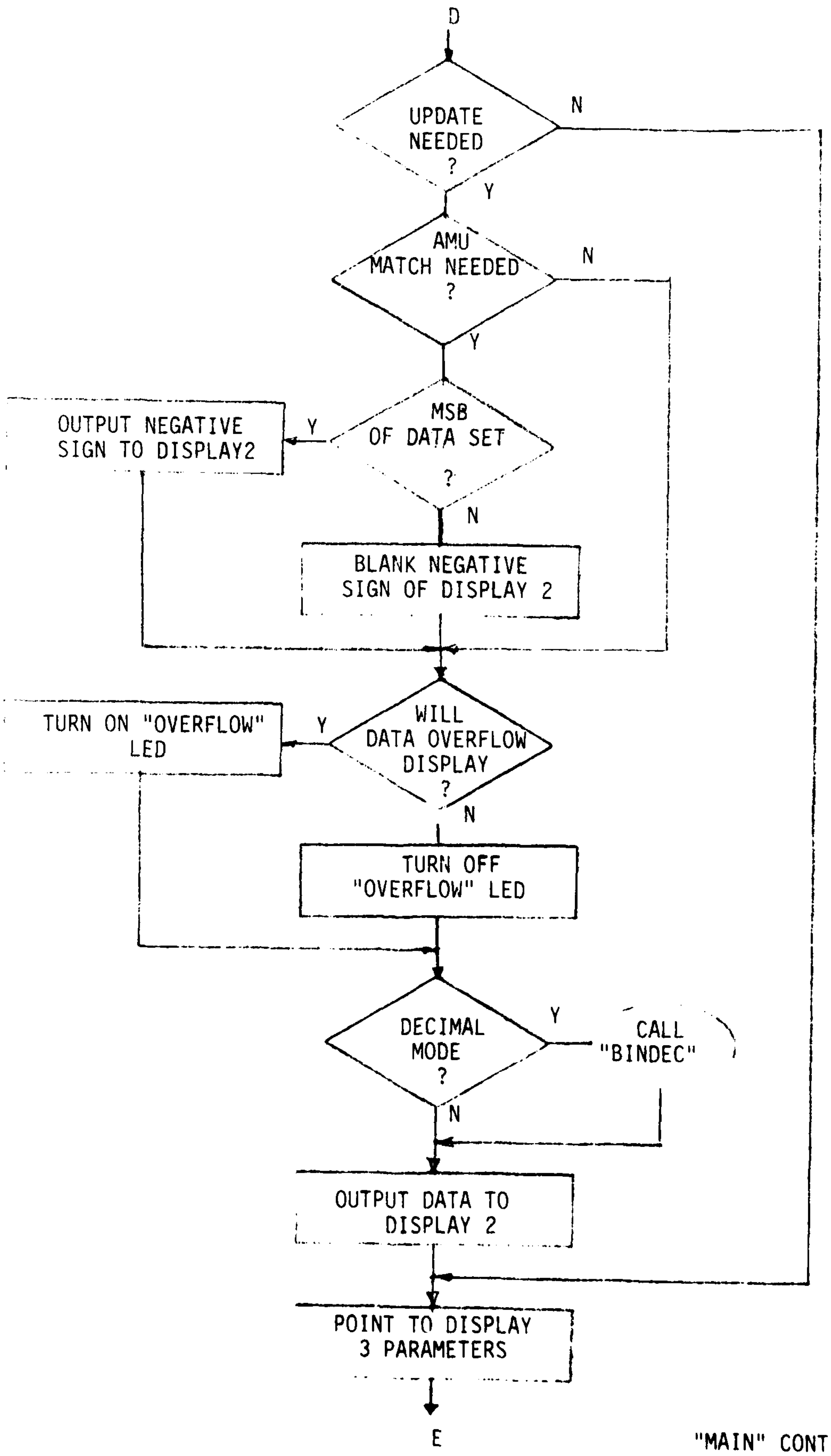

BEGIN
CLEAR "BUFFER FULL" FLAGS
CALL "FRAME"
A
CALL "LDBUFF"
N
IN SYNC ?
N
TURN OFF SYNC LED
OUT OF SYNC TWICE ?
Y
N
Y
TURN ON SYCN LED
DATA VALID ?
N
Y
OUTPUT AMU AND DATA TO THE DAC'S
DATA NEGATIVE ?
Y
TURN ON "NEGATIVE" LED
N
TURN OFF "NEGATIVE" LED
B
"MAIN"


B
BURST READY FLAG SET ?
Y
BUZZ
N
DO NOT BUZZ
DATA VALID ?
Y
TURN ON "DATA VALID" LED
N
TURN OFF" DATA VALID" LED
TTY ACTIVE FLAG SET ?
Y
JUMP INTO "TTYLNK"
N
RESET AMU DECOM MATCH FLAG
DATA VALID ?
N
Y
AMU MATCH PRESENT ?
Y
SET AMU MATCH FLAG
N
POINT TO DISPLAY ONE PARAMETERS
C

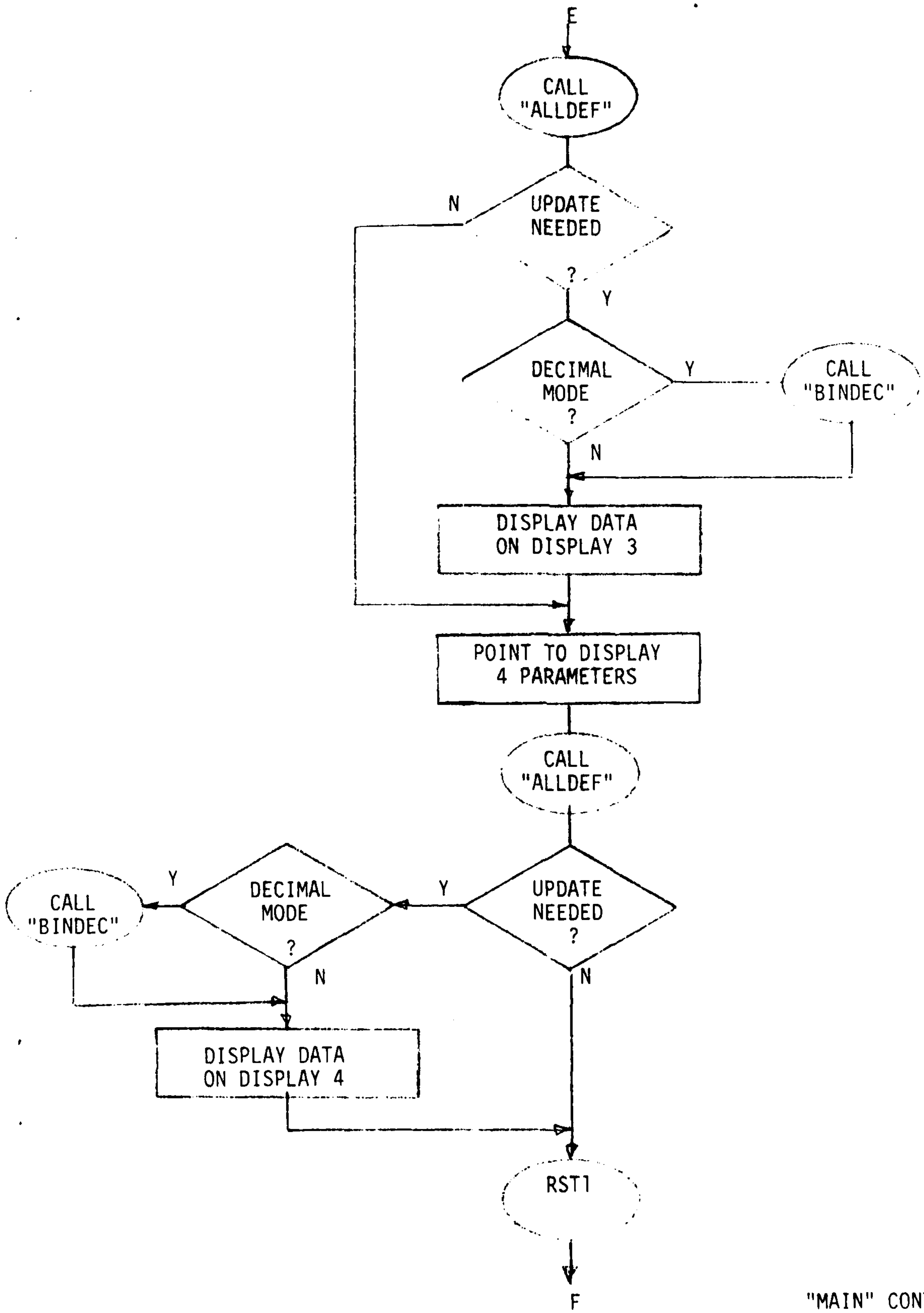
"MAIN" CONT.


C
CALL "ALLDEF"
UP-DATE NEEDED ?
N
Y
AMU MATCH FLAG SET ?
N
Y
DIVIDE AMU BY 4
CALL "BINDEC"
Y
DECIMAL MODE ?
N
DISPLAY IN DISPLAY 1
POINT TO DISPLAY 2 PARAMETERS
CALL "ALLDEF"
D

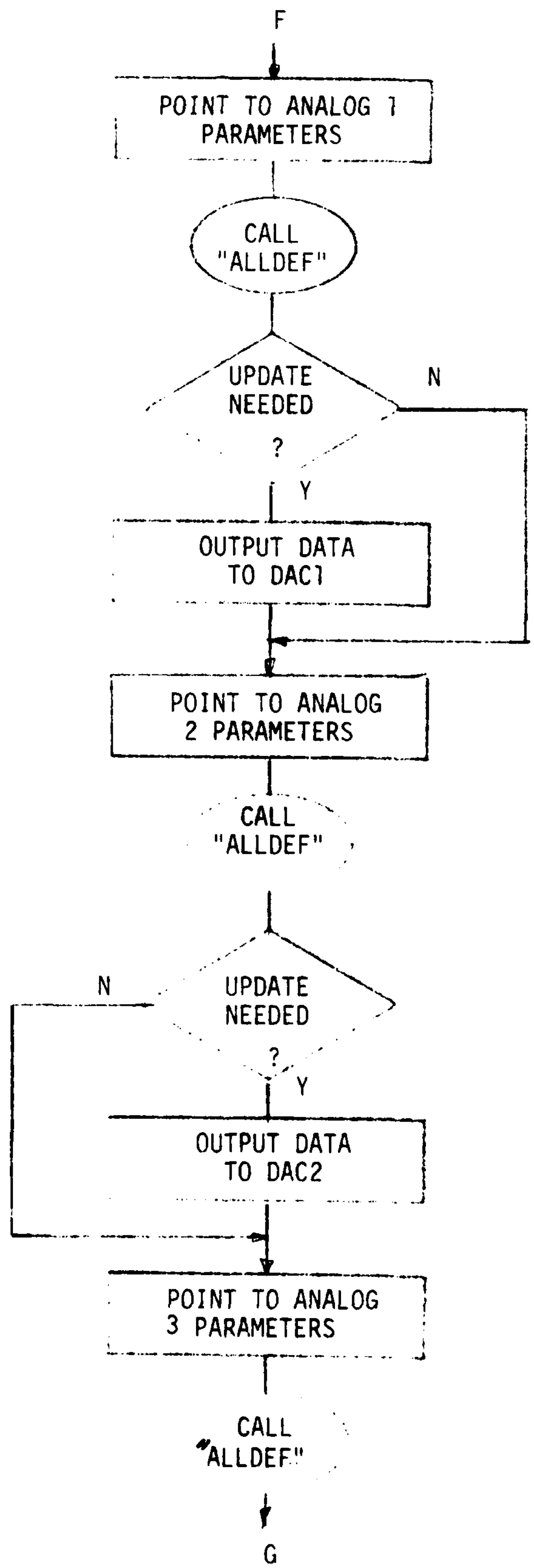
"MAIN" CONT.


D
UPDATE NEEDED ?
N
Y
AMU MATCH NEEDED ?
N
Y
OUTPUT NEGATIVE SIGN TO DISPLAY2
Y
MSB OF DATA SET ?
N
BLANK NEGATIVE SIGN OF DISPLAY 2
TURN ON "OVERFLOW" LED
Y
WILL DATA OVERFLOW DISPLAY ?
N
TURN OFF "OVERFLOW" LED
DECIMAL MODE ?
Y
CALL "BINDEC"
N
OUTPUT DATA TO DISPLAY 2
POINT TO DISPLAY 3 PARAMETERS
E

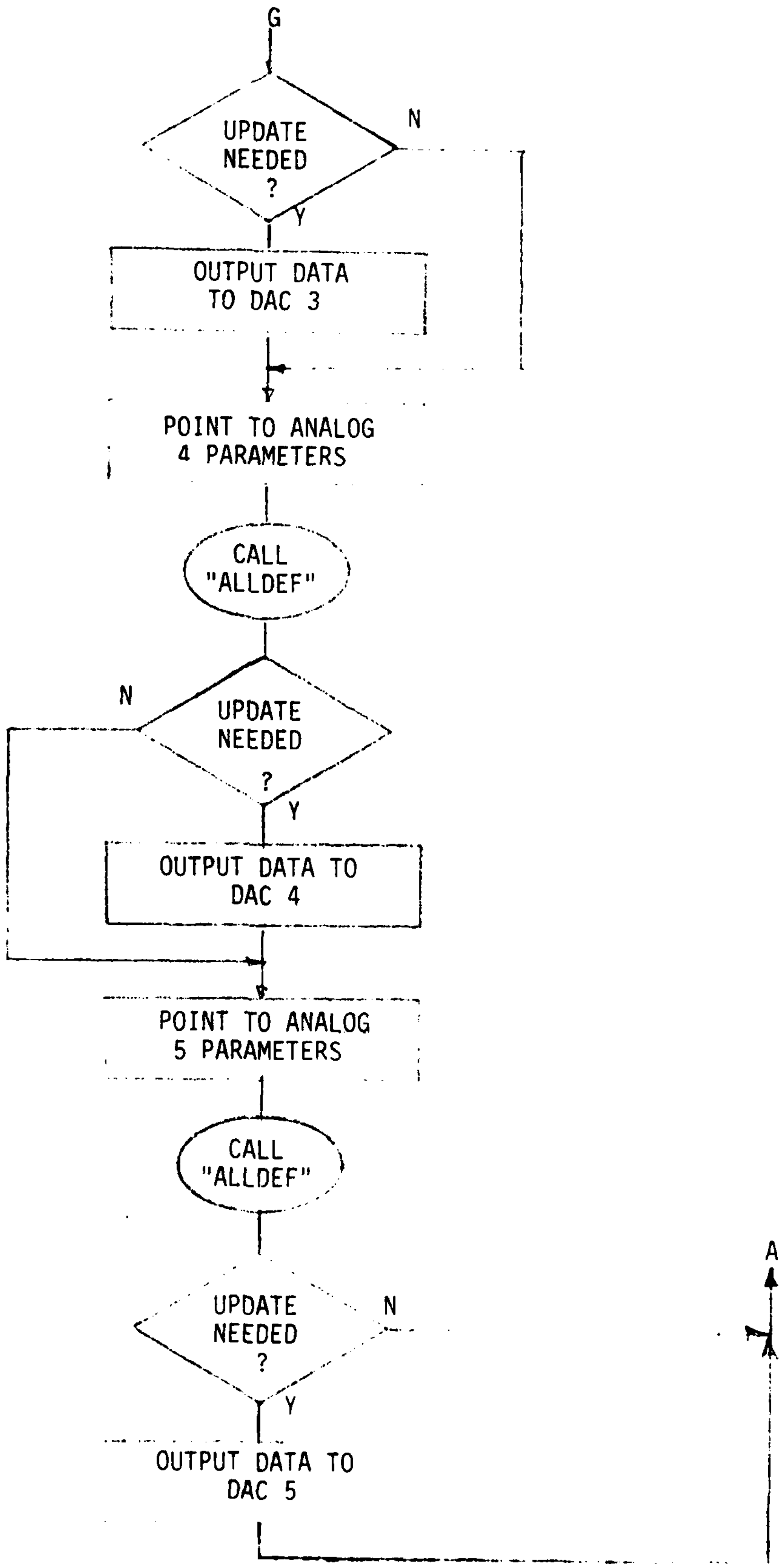
"MAIN" CONT.


E
CALL "ALLDEF"
UPDATE NEEDED ?
N
Y
DECIMAL MODE ?
Y
CALL "BINDEC"
N
DISPLAY DATA ON DISPLAY 3
POINT TO DISPLAY 4 PARAMETERS
CALL "ALLDEF"
UPDATE NEEDED ?
Y
DECIMAL MODE ?
Y
CALL "BINDEC"
N
DISPLAY DATA ON DISPLAY 4
N
RST1
F

"MAIN" CONT.


F
POINT TO ANALOG 1 PARAMETERS
CALL "ALLDEF"
UPDATE NEEDED ?
N
Y
OUTPUT DATA TO DAC1
POINT TO ANALOG 2 PARAMETERS
CALL "ALLDEF"
N
UPDATE NEEDED ?
Y
OUTPUT DATA TO DAC2
POINT TO ANALOG 3 PARAMETERS
CALL "ALLDEF"
G

"MAIN" CONT.


G
UPDATE NEEDED ?
N
Y
OUTPUT DATA TO DAC 3
POINT TO ANALOG 4 PARAMETERS
CALL "ALLDEF"
N
UPDATE NEEDED ?
Y
OUTPUT DATA TO DAC 4
POINT TO ANALOG 5 PARAMETERS
CALL "ALLDEF"
A
UPDATE NEEDED ?
N
Y
OUTPUT DATA TO DAC 5

RST1

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: REGISTERS NOT AFFECTED.

MEMORY: 3800 → MAX MEMORY ;LOOK UP TABLES.

1818 → 27 ;DISPLAY BUFFER.

I/O PORTS: 81 ;DISPLAY CONTROLLER COMMAND.

80 ;DISPLAY CONTROLLER.

INT75

TYPE: INTERUPT.

ENTER: NO CONDITIONS.

RETURN: REGISTERS NOT AFFECTED.

COMMENT: INT 7.5 MUST BE MASKED, AND INTERUPTS MUST BE ENABLED.

MEMORY:	18E0-1	;CONTAINS BUFFER POINTER ADDRESS.
	1860-DF	;TM BUFFER.
	18E2	;BYTES IN TM FRAME.
	18E3	;"TM BUFFER FULL" FLAGS.

I/O PORTS:	A0	;PCM INPUT
	A1	;PCM MONITOR
	A3	;MONITOR STOBE.

CMBACK

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: NO RETURN, JUMPS TO CALLED PROGRAM.
SP IS POINTING AT "CMBACK."

MEMORY:	1848	;ADDRESS STATUS FLAG.
	18EA	;"TRMIN" FLAG.

FRAME

```
    TYPE:  SUBROUTINE.

    ENTER:  NO CONDITIONS.

    RETURN:    PSW AFFECTED.
               BC AFFECTED.
               H AFFECTED.
               INTERUPTS ENABLED.
               INT 7.5 ON

    MEMORY:  18E2                       ;TM FRAME LENGTH.

    I/O PORTS:  A0                      ;PCM INPUT.
                A3                      ;INTERUPT CLOCK RESET.
```

YORN

```
    TYPE:  SUBROUTINE.

    ENTER:  NO CONDITIONS.

    RETURN:    CY=1 FOR "NO" ENTRY.
               CY=0 FOR "YES" ENTRY.
               PSW AFFECTED.

    MEMORY:  182A-B                     ;"ESCAPE" ADDRESS.
```

MODE

```
    TYPE: PROGRAM.

    ENTER:  NO CONDITIONS.

    RETURN:  ALL AFFECTED.

    MEMORY:    MESSAGE  33
               MESSAGE  34
               MESSAGE  35
               MESSAGE  36
               MESSAGE  37
               MESSAGE  41
               MESSAGE  42
               2409                     ;MODE WORD FOR BALLOON.
               19C4-5                   ;MODE WORD (ASCII) FOR BALLOON.
               2415                     ;STEPPING VALUE FOR BALLOON.
               19DC-D                   ;STEPPING VALUE (ASCII) FOR BALLOON.
```

BIASP

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

```
MEMORY:  MESSAGE  38
          3805                    ;SIXTH DISPLAY DIGIT, HEXIDECIMAL.
          240A-E                  ;PRIMARY BIASES FOR BALLOON.
          19C4-D                  ;PRIMARY BIASES (ASCII) FOR BALLOON.
```

BIASS

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

```
MEMORY:  MESSAGE  38
          3805                    ;SIXTH DISPLAY DIGIT, HEXADECIMAL.
          240F-13                 ;SECONDARY BIASES FOR BALLOON.
          19CE-D7                 ;SECONDARY BIASES (ASCII) FOR BALLOON.
```

READ

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: FLAGS AFFECTED.
ACC CONTAINS INPUT ENTRY.

```
MEMORY:  1855                     ;KEYBOARD/TERMINAL FLAG.
         184D                     ;CLEAR DISPLAY FLAG
         18EA                     ;ASCII CONVERT FLAG.

I/O PORTS:  91                    ;USART COMMAND.
            90                    ;USART.
            81                    ;KEYBOARD COMMAND.
            80                    ;KEYBOARD.
```

NMREAD

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: ACC CONTAINS NUMBER.
FLAGS AFFECTED.

MEMORY:	1818-1D	;HEXADECIMAL DISPLAY BUFFER.
	182A-B	;ALL PURPOSE ESCAPE ADDRESS.

MOVE

TYPE: SUBROUTINE.

ENTER: HL CONTAINS BEGINNING OF BLOCK TO BE MOVED.
DE CONTAINS END OF BLOCK TO BE MOVED.
BC CONTAINS BEGINNING OF BLOCK TO BE MOVED TO.

RETURN: HL EQUAL TO DE.
BC CONTAINS LAST LOCATION OF BLOCK TO BE MOVED TO.
PSW AFFECTED.

CMPDH

TYPE: SUBROUTINE.

ENTER: HL CONTAINS DATA TO BE COMPARED TO DE.
DE CONTAINS DATA TO BE COMPARED TO HL.

RETURN: ACC AFFECTED.
CY=1 Z=0 HL > DE
CY=0 Z=0 HL < DE
CY=0 Z=1 HL = DE

ALREAD

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: ACC CONTAINS ALFA-NUMERIC CHARACTER.
FLAGS AFFECTED.

MEMORY:	1810-7	;ALFA-NUMERIC DIDSPLAY BUFFER.
	182A-B	;EXECUTIVE JUMP ADDRESS.

```
BELL

    TYPE:  SUBROUTINE.

    ENTER:  HL CONTAINS RELATIVE TIME FOR BELL TO RING.

    RETURN:  HL AFFECTED.

    I/O PORTS:  B3                    ;TURN ON FOR BELL.

DIRECT

    TYPE:  PROGRAM.

    ENTER:  NO CONDITONS.

    RETURN:  ALL REGISTERS AFFECTED.

    MEMORY:  1810                     ;LAST ENTRY OF DISPLAY BUFFER.
             1800-F                   ;STATUS LOCATIONS.

ERROR

    TYPE:  SUBROUTINE.

    ENTER:  NO CONDITIONS.

    RETURN:  HL AFFECTED.
             PSW AFFECTED.

    MEMORY:  MESSAGE 9.

ENDIT

    TYPE:  SUBROUTINE.

    ENTER:  NO CONDITIONS.

    RETURN:  PSW AFFECTED.
             HL AFFECTED.

    MEMORY:  MESSAGE 10.

MESSAGE

    TYPE:  SUBROUTINE.

    ENTER:  HL POINTS TO BEGINNING OF ASCII STRING.
            ACC CONTAINS NUMBER OF CHARACTERS.

    RETURN:  HL AFFECTED.
             PSW AFFECTED.

    MEMORY:  3800-3807                ;ALPHA-NUMERIC DISPLAY.
```

FILL

TYPE: SUBROUTINE.

ENTER: HL CONTAINS BEGINNING OF BLOCK TO BE FILLED.
DE CONTAINS END OF BLOCK TO BE FILLED.
ACC CONTAINS DATA TO BE PUT INTO BLOCK.

RETURN: PSW AFFECTED.
HL EQUAL TO DE.

MEMORY: 184C ;TEMP. BUFFER.

CLEARB

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS OK.

I/O PORTS: 81 ;HEXIDECIMAL DISPLAY COMMAND.

CLEAR

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS OK.

MEMORY: 3800-7 ;ALPHA-NUMERIC DISPLAY.
1810-27 ;ALL DISPLAY BUFFERS.

I/O PORTS: 81 ;HEXIDECIMAL DISPLAY COMMAND.

MANY

TYPE: SUBROUTINE.

ENTER: NO CONDITIONS.

RETURN: B CONTAINS NUMBER OF ENTRIES.
PSW AFFECTED.

MEMORY: 1855 ;GCU/TERMINAL FLAG.
184D ;"CLEAR: FLAG.

```
LOOKUP

    TYPE:   SUBROUTINE.

    ENTER:  NO CONDITIONS.

    RETURN:  IF CY=0
                THEN  HL CONTAINS ADDRESS OF ROUTINE.
                      DE CONTAINS ROUTINES STATUS ADDRESS.
                      ACC CLEARED.
                      FLAGS AFFECTED.
                      BC AFFECTED.

             IF CY=1
                THEN ALL REGISTERS AFFECTED.

    MEMORY:  ALL LOOKUP TABLE
             1810-7                     ;ALFA-NUMERIC DISPLAY BUFFER.
             1855                       ;"TERMINAL/GCU" FLAG.
             3800-7                     ;ALFA-NUMERIC DISPLAY.

ADDRES

    TYPE:   SUBROUTINE.

    ENTER:  NO CONDITIONS.

    RETURN:  IF CY=0
                THEN DE CONTAINS ADDRESS.
                  ALL OTHER REGISTERS AFFECTED.

             IF CY=1,   B=5
                THEN MODE ERROR, ALL REGISTERS AFFECTED.

             IF CY=1  B 5
                THEN GENERAL ERROR, ALL REGISTERS AFFECTED.

    MEMORY:  1818-F                     ;ALFA-NUMERIC DISPLAY BUFFER.
             1848                       ;"ADDRESS/DATA" FLAG.

INDRCT

    TYPE:   PROGRAM.

    ENTER:  NO CONDITIONS.

    RETURN:  ALL REGISTERS AFFECTED.

    MEMORY:  1810                       ;LAST ENTRY INTO ALFA-NUMERIC DISPALY BUFFER.
             1800-F                     ;STATUS LOCATIONS.
```

BNRY

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

```
MEMORY:  1810                        ;LAST ENTRY INTO ALFA-NUMERIC DISPLAY BUFFER.
         1800-F                      ;STATUS LOCATIONS.
```

DCML

TYPE: PROGRAM.

ENTER: NO CONDITONS.

RETURN: ALL REGISTERS EFFECTED.

```
MEMORY:  1810                        ;LAST ENTRY INTO ALFA-NUMERIC DISPLAY BUFFER
         1800-F                      ;STATUS LOCATIONS.
```

MOVEM

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

```
MEMORY:   MESSAGE 1.
          MESSAGE 2.
          MESSAGE 3.
          MESSAGE 4.
          182A-B                     ;ALL PURPOSE ESCAPE LOCATION.
```

COMPA

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

```
MEMORY:   184E                       ;"MEMORY VERSUS MEMORY" FLAG.
          MESSAGE 6.
          MESSAGE 11.
          MESSAGE 4.
          182A-B                     ;ALL PURPOSE ESCAPE LOCATION.
          MESSAGE 5.
```

COMPD

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

```
MEMORY:  MESSAGE 4.
         MESSAGE 5.
         MESSAGE 6.
         MESSAGE 11.
         184E                  ;"MEMORY VERSES MEMORY" FLAG.
         182A-B                ;ALL PURPOSE ESCAPE LOCATION.
```

COMPSB

TYPE: SUBROUTINE.

ENTER: ACC CONTAINS DATA TO BE DISPLAYED.
BC CONTAINS ADDRESS TO BE DISPLAYED.

RETURN: ALL REGISTERS OK.

ALTR

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

```
MEMORY:  MESSAGE 5.
         1849                  ;"RETURN ON CR" FLAG.
         182A-B                ;ALL PURPOSE ESCAPE ADDRESS.
         MESSAGE 9.
```

GETDAT

TYPE: SUBROUTINE.

ENTER: NO CONDITONS.

RETURN: IF CY=1, B=5
THEN MODE ERROR, ALL REGISTERS AFFECTED EXCEPT HL.

IF CY=1, B 5
THEN GENERAL ERROR.

IF CY=0
THEN ACC CONTAINS DATA.
HL OK.
ALL OTHER REGISTERS AFFECTED.

MEMORY:	1848	;"ADDRESS/DATA" FLAG.
	MESSAGE 7.	
	1849	;"RETURN ON CR" FLAG.

DECBIN

TYPE: SUBROUTINE.

ENTER: HL CONTAINS DECIMAL NUMBER.

RETURN: HL CONTAINS BINARY NUMBER.

MEMORY:	184A-B	;TEMP. BINARY EQUIVALENT STORAGE.
	184F	;TEMP. STORAGE OF"BIN"LSBYTE.
	"BIN"-END OF "BIN"	;POINTS TO DECIMAL EQUIVALENCE
	MESSAGE 9.	
	182A-B	;ALL PURPOSE ESCAPE ADDRESS.

DISPL

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

MEMORY:	1855	;"CONSOLE/TERMINAL" FLAG.
	MESSAGE 6.	
	182A-B	;ALL PURPOSE ESCAPE ADDRESS.
	MESSAGE 5.	
	MESSAGE 4.	
I/O PORTS:	91	;USART COMMAND.
	90	;USART.

```
FEPROM

     TYPE: PROGRAM.

     ENTER:  NO CONDITIONS.

     RETURN:  ALL REGISTERS AFFECTED.

     MEMORY:   2400-5FF                ;CONTAINS DATA FOR FAKE EPROM.
               FE00-FFFF               ;FAKE EPROM.

     I/O PORTS: A3                     ;"EPROM/RAM" FLAG.

TTYLNK

     TYPE:  SUBROUTINE.

     ENTER:  NO CONDITIONS.

     RETURN:  C IS OK.
              ALL OTHER REGISTERS AFFECTED.

     MEMORY:   1828                    ;LOOP COUNTER.
               1829                    ;"USART/PCM LINK" FLAG.
               19AE-F                  ;ADDRESS OF TTY DATA.
               19B0-DF                 ;DATA TO BE SENT TO BALLOON.

     I/O PORTS:  D3                    ;COMMAND FOR "TIME-OUT" TIMER.
                 D2                    ;"TIME-OUT" TIMER.
                 90                    ;SYSTEM USART.
                 C1                    ;TM USART COMMAND.
                 CO                    ;TM USART.

BINDEC

     TYPE:  SUBROUTINE.

     ENTER:  DE CONTAINS BINARY DATA.

     RETURN:  DE CONTAINS DECIMAL EQUIVELENT

     MEMORY:   184F                    ;BIT COUNTER.
               182C                    ;"MSBYTE PROCEESSED" FLAG.
               "BIN"-END OF "BIN"      ;DECIMAL EQUIVALENCES.
```

GOTO

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: DOES NOT RETURN.

COMMENT: PC IS MODIFIED.

MEMORY: MESSAGE 5.

DSDATA

TYPE: SUBROUTINE.

ENTER: ACC CONTAINS DATA TO BE DISPLAYED.
HL CONTAINS STATUS ADDRESS OF CALLING ROUTINE.

RETRUN: NO REGISTERS AFFECTED.

MEMORY: STATUS OF CALLING ROUTINE.
3805-7 ;ALFA-NUMERIC DISPLAY.

DSADDR

TYPE: SUBROUTINE.

ENTER: DE CONTAINS ADDRESS TO BE DISPLAYED.
HL CONTAINS STATUS ADDRESS OF CALLING ROUTINE.

RETURN: NO REGISTERS AFFECTED.

MEMORY: STATUS OF CALLING ROUTINE.
3800-3803 ;ALFA-NUMERIC DISPLAY.

BAUD

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

MEMORY: MESSAGE 13
1850 ;FAKE STATUS ADDRESS.

I/O PORTS: D3 ;SYSTEM USART CLOCK COMMAND.
D0 ;SYSTEM USART CLOCK.

RPAGE

TYPE: PROGRAM

ENTER: NO CONDITONS.

RETURN: DOES NOT RETURN UNLESS IN ERROR: THEN ALL REGISTERS AFFECTED.

COMMENT: JUMPS INTO MAIN ROUTINE.

MEMORY:	182F	; "PAGE/BOOK" FLAG.
	MESSAGE 12A	
	184F	;TEMP. FAKE STATUS.
	19BØ-2	;LOCATIONS OF BALLOON TM BUFFER.

RBOOK

TYPE: PROGRAM

ENTER: NO CONDITIONS.

RETURN: DOES NOT RETURN UNLESS IN ERROR; THEN ALL AFFECTED.

COMMENT: JUMPS INTO MAIN ROUTINE.

MEMORY:	182F	;PAGE/BOOK: FLAG.
	MESSAGE 12A	
	184F	;TEMP. FAKE STATUS.
	19BØ-2	;LOCATIONS OF BALLOON TM BUFFER.

NPAGE

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: DOES NOT RETURN.

COMMNET: JUMPS INTO MAIN ROUTINE.

MEMORY: 19BØ-2 ;LOCATIONS OF BALLOON TM BUFFER.

DUMP

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: DOES NOT RETURN.

COMMENT: JUMPS INTO MAIN ROUTINE.

MEMORY: 19BØ-2 ;LOCATIONS OF BALLOON TM BUFFER.

CONT

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: DOES NOT RETURN.

COMMENT:: JUMPS INTO MAIN ROUTINE.

MEMORY: 19BØ-2 ;LOCATIONS OF BALLOON TM BUFFER.

WAIT

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: DOES NOT RETURN.

COMMENT: JUMPS INTO MAIN ROUTINE.

MEMORY: 19BØ-2 ;LOCATIONS OF BALLOON TM BUFFER.

RATIO

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS.

MEMORY: MESSAGE 31.
1809 ;CONTAINS STATUS.
2407-8 ;STORAGE OF DATA FOR USER.

MASK

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

MEMORY: MESSAGE 31.
2414 ;STORAGE OF DATA FOR USER.

IDNUM

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

```
MEMORY:  MESSAGE 31.
         2416-7                  ;STORAGE OF DATA FOR USER.
```

LOP

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

```
MEMORY:  MESSAGE 31.
         2404                    ;STORAGE OF DATA FOR USER.
```

TIME

TYPE: PROGRAM.

ENTER: NO CONDITONS.

RETURN: ALL REGISTERS AFFECTED.

```
MEMORY:  MESSAGE 31.
         1809                    ;CONTAINS STATUS.
         2405-6                  ;STORAGE OF DATA FOR USER.
```

AMU

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

```
MEMORY:  MESSAGE 30.
         MESSAGE 31.
         MESSAGE 32.
         1809                    ;CONTAINS STATUS.
         2400-3                  ,STORAGE OF DATA FOR USER.
```

INITAL

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

```
MEMORY:  18EB            ;"TWO BYTE DEFINITION" FLAG.
         MESSAGE 22.
         18E4-9          ;STORAGE OF ADDRESSES FOR TM DATA.
         MESSAGE 21.
         MESSAGE 23.
         MESSAGE 24.
         3806-7          ;ALFA-NUMERIC DISPLAY.
         1908-F          ;STORAGE OF ADDRESS FOR TM DATA.
         MESSAGE 25.
         1809            ;CONTAINS STATUS.
         MESSAGE 26.
         1916-9          ;STORAGE OF ADDRESSES FOR TM DATA.
         1922-5          ;STORAGE OF ADDRESSES FOR TM DATA.
         192E-31         ;STORAGE OF ADDRESSES FOR TM DATA.
         MESSAGE 28.
         196A-B          ;STORAGE OF ADDRESS FOR TM DATA.
         197A-B          ;STORAGE OF ADDRESS FOR TM DATA.
         1946-7          ;STORAGE OF ADDRESS FOR TM DATA.
         1952-3          ;STORAGE OF ADDRESS FOR TM DATA.
         195E-F          ;STORAGE OF ADDRESS FOR TM DATA.
         MESSAGE 29.
         1849            :"ADDRESS/DATA" FLAG.
         19AE-F          ;STORAGE OF ADDRESS FOR TM DATA.
         MESSAGE 39.
         1829            ;"TM/USART" FLAG.
```

GETLOC

TYPE: SUBROUTINE

```
ENTER:  HL CONTAINS ADDRESS OF MESSAGE.
        ACC CONTAINS LENGTH OF MESSAGE.

RETURN:  IF CY=1
           THEN ERROR HAS OCCURRED.
          ALL REGISTERS AFFECTED.
         IF CY=0
           THEN ACC CONTAINS LOBYTE OF ADDRESS.
           HL AFFECTED.
           FLAGS AFFECTED.
```

SWITCH

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

MEMORY: 1855 ;"TERMINAL/CONSOLE": FLAG.

I/O PORTS: B[illegible] ;CONSOLE/TERMINAL LED MONITOR
B3 ;CONSOLE/TERMINAL LED

ASCONV

TYPE: SUBROUTINE.

ENTER: ACC CONTINAS ASCII CHARACTER TO BE CONVERTED INTO SYSTEM BINARY.

RETURN: FLAGS AFFECTED.
ACC CONTAINS SYSTEM BINARY.

BINCON

TYPE: SUBROUTINE.

ENTER: ACC CONTAINS SYSTEM BINARY TO BE CONVERTED INTO ASCII.

RETURN: FLAGS AFFECTED.

ACC CONTAINS ASCII

FILLM

TYPE: PROGRAM.

ENTER: NO CONDITIONS.

RETURN: ALL REGISTERS AFFECTED.

MEMORY: MESSAGE 4.
MESSAGE 5.
MESSAGE 6.
MESSAGE 7.
182A-B ;ALL PURPOSE ESCAPE ADDRESS.

TRMOUT

TYPE: SUBROUTINE.

ENTER: ACC CONTAINS DATA TO BE TRANSMITTED.

RETURN: ALL REGISTERS OK.

I/O PORTS: 91 ;SYSTEM USART COMMAND.
90 ;SYSTEM USART.

TRNSMT

TYPE: SUBROUTINE.

ENTER: ACC CONTAINS DATA TO BE SENT TO BALLOON.

RETURN: FLAGS AFFECTED.

I/O PORTS: C1 ;TM USART COMMAND.
C0 ;TM USART.

[illegible]

    TYPE: SUBROUTINE.

    ENTER: NO CONDITIONS.

    RETURN: IF CY=1
               THEN EITHER OUT OF SYNC OR OUT OF TIME.
               PSW AFFECTED.
             IF CY=0
               THEN ACC CONTAINS DATA.
               FLAGS AFFECTED.

    MEMORY: 1809   ;"PCM/USART" FLAG.
             196E   ;STATUS BYTE FROM BALLOON.
             19AE   ;CONTAINS ADDRESS OF TTY BYTE.

    I/O PORTS: [illegible]   ;"TIME-OUT" FLAG.
                C1   ;TM USART COMMAND.
                C0   ;TM USART.

CNVRT
    TYPE: SUBROUTINE.

    ENTER: HL CONTAINS LOCATION WHERE MSNIBBLE IS TO BE STORED.
            DE CONTAINS DATA TO BE CONVERTED INTO 4 ASCII CHARACTERS.

    RETURN: HL CONTAINS LOCATION WHERE NEXT MSNIBBLE IS TO BE STORED.
             PSW AFFECTED.

    MEMORY: (HL) - [(HL) + 4]   ;ASCII STORAGE.

CNVRT

    TYPE: SUBROUTINE.

    ENTER: E CONTAINS DATA TO BE CONVERTED INTO 2 ASCII CHARACTERS.
            HL CONTAINS LOCATION OF WHERE MS NIBBLE IS TO BE STORED.

    RETURN: PSW AFFECTED.
             HL CONTAINS LOCATION WHERE NEXT MS NIBBLE IS TO BE STORED.

    MEMORY: (HL) - [(HL) +2]

```
GETALL

     TYPE:   SUBROUTINE.

     ENTER:  DE CONTAINS STARTING ADDRESS OF PARAMETER LIST.

     RETURN:  IF CY=1
                THEN REGISTERS HOLD NO VALID DATA.
              IF CY=0
                THEN  DE CONTAINS ADDRESS OF  HIGH BYTE.
                      HL CONTAINS ADDRESS OF LOW BYTE.
                      PSW AFFECTED.
                      BC AFFECTED.

     MEMORY:  MESSAGE 15.
              MESSAGE 16.
              MESSAGE 40.
              MESSAGE 17.
              MESSAGE 18.
              1849                  ;"RETURN ON CR" FLAG.
              18EB                  ;"ANALOG DEFINITION" FLAG.
              MESSAGE 19.
              MESSAGE 20.
              182A                  ;ALL PURPOSE ESCAPE ADDRESS.
              (DE) AND UP           ;PARAMETERS FOR DISPLAYING.

ALLDEF

     TYPE:   SUBROUTINE.

     ENTER:  HL CONTAINS POINTER POINTING TO TOP OF PARAMETER LIST FOR DISPLAYING.

     RETURN:  IF CY=1
                THEN PSW AFFECTED.
              IF CY=0
                THEN HL CONTAINS DATA TO BE DISPLAYED.
                PSW AFFECTED.

     MEMORY:  (HL) AND UP           ;PARAMETERS FOR DISPLAYING.
```

```
[illegible]

     TYPE:   SUBROUTINE.

     ENTER.  NO CONDITIONS.

     RETURN:  HL CONTAINS ADDRESS OF FIRST BYTE OF SYNC WORD.
              ALL OTHER REGISTERS AFFECTED.

     MEMORY:  18E3                   ;"BUFFER FULL" FLAGS.
              18E2                   ;NUMBER OF WORDS IN TM FRAME.

MAIN

     TYPE:   PROGRAM.

     ENTER:  NO CONDITIONS.

     RETURN:  ALL REGISTERS AFFECTED.

     MEMORY:  18E3                   ;"BUFFER FULL" FLAGS.
              18E0-1                 ;CONTAINS BEGINNING OF TM BUFFER.
              196E                   ;PCM STATUS WORD FROM TM BUFFER.
              18E4-5                 ;AMU NUMBER FROM TM BUFFER.
              18E6-7                 ;DATA FROM TM BUFFER.
              190C-D                 ;AMU MATCH.
              1830                   ;"MATCH" FLAG.
              1900                   ;FLAGS FOR DISPLAY.
              1810-1B                ;DISPLAY1 BUFFER.
              190E                   ;FLAGS FOR DISPLAY2.
              1823-27                ;DISPLAY2 BUFFER.
              191A                   ;FLAGS FOR DISPLAY3.
              181E-21                ;DISPLAY3 BUFFER.
              1926                   ;FLAGS FOR DISPLAY.
              3800-3                 ;ALFA-NUMERIC DISPLAY.
              18E8-9                 ;ADDRESS OF PRESENT SUBID NUMBER.

     I/O PORTS:

              B3                     ;SYNC LED.
              F0                     ;ANALOG CHANNEL 1.
              F1                     ;ANALOG CHANNEL 2.
              F2                     ;ANALOG CHANNEL 3.
              F3                     ;ANALOG CHANNEL 4.
              F4                     ;ANALOG CHANNEL 5.
              F5                     ;LEDS.
              F6                     ;ANALOG CHANNEL 7.
              F7                     ;ANALOG CHANNEL 8.
              F8                     ;ANALOG CHANNEL 9.
              F9                     ;ANALOG CHANNEL 10.
```

EPROM

0-17FF, 2800-2FFF

I/PORTS

8Ø	HEXDISPLAY AND KEYBOARD	;8279A.
81	COMMAND FOR ABOVE	;8279A.
9Ø	SYSTEM USART	;8251A.
91	COMMAND FOR ABOVE	;8251A.
AØ	PCM DATA	;8251A.
A1	TM MONITOR	;8255A.
A2	FUNCTION BITS	;8255A.
A3	COMMAND FOR ALL 3 ABOVE	;8255A.
BØ	SPARE	;8255A.
B1	SPARE	;8255A.
B2	LEDS	;8255A.
B3	COMMAND FOR ALL 3 ABOVE	;8255A.
CØ	TM USART	;8251A.
C1	COMMAND FOR ABOVE	;8251A.
DØ	SYSTEM USART CLOCK	;8253-5.
D1	TM USART CLOCK	;8253-5.
D2	TIME-OUT TIMER	;8253-5.
D3	COMMAND FOR ALL 3 ABOVE	;8253-5.
FØ	ANALOG MSBYTE AMU.	
F1	ANALOG LSBYTE AMU.	
F2	ANALOG MSBYTE DATA.	
F3	AMALOG NSBYTE DATA.	
F4	ANALOG LSBYTE DATA.	
F5	ANALOG CHANNEL 1.	
F6	ANALOG CHANNEL 2.	
F7	ANALOG CHANNEL 3.	
F8	ANALOG CHANNEL 4.	
F9	ANALOG CHANNEL 5.	
FA	SPARE.	
FB	SPARE.	
FC	SPARE.	
FD	SPARE.	
FE	FAKE EPROM 1.	
FF	FAKE EPROM 2.	

RAM

RAM EXTENDS FROM 1800H to 27FFH.

1800-180F	FLAGS FOR MODE OF FUNCTIONS.
1810-1827	KEYBOARD ENTRY BUFFER.
1828	LOOP COUNTER FOR BALLOON RECEIVER.
1829	"USART/TM" FLAG.
182A-182B	ALL PURPOSE ESCAPE LOCATION.
182C	"MSBYTE HAS BEEN PROCESSED" FLAG.
182D-182E	SPARES.
182F	"INSTRUCTION SET OR PROGRAM" FLAG.
1830	"MATCH" FLAG.
1831-1847	SPARES

1848	"ADDRESS" FLAG.
1849	"RETURN ON CR" FLAG.
184A-184B	BINARY EQUIVALENT (DECBIN).
184C	BUFFER FOR "FILL".
184D	"CLEAR DISPLAY" FLAG.
184E	"COMPA/COMPD" FLAG.
184F	COUNTER OF BITS (BINDEC).
1850	FAKE STATUS (BAUD).
1851-1854	SPARE.
1855	"TERM/GCU" FLAG.
1856-185F	SPARE.
1860-189E	TM DATA BUFFER #1.
18A0-18DF	TM DATA BUFFER #2.
18E0-18E1	TM BUFFER POINTER.
18E2	TM FRAME LENGTH.
18E3	"BUFFER FALL" FLAGS,
18E4-18E5	AMU POINTER FOR D/A CONVERSION.
18E6-18E7	DATA POINTER FOR D/A CONVERSION.
18E8-18E9	SUB-ID POINTER.
18EA	"NO ASCII TO SYSTEM BINARY" FLAG.
18EB	"HIBYTE LOCATION ONLY" FLAGS.
18EC-18FF	SPARE.
1900	DISPLAY 1 FLAGS.
1901-1907	SUB-ID NUMBERS TO BE MATCHED.
1908-1909	LOCATION OF BYTE1 LOW.
190A-190B	LOCATION OF BYTE2 HIGH.
190C-190D	AMU MATCH DATA.
190E	DISPLAY 2 FLAGS.
190F-1915	SUB-ID NUMBERS TO BE MATCHED.
1916-1917	LOCATION OF BYTE1 LOW.
1918-1919	LOCATION OF BYTE2 HIGH.
191A	DISPLAY 3 FLAGS.
191B-1921	SUB-ID NUMBERS.
1922-1923	LOCATION OF BYTE1 LOW.
1924-1925	LOCATION OF BYTE2 HIGH.
1926	DISPLAY 4 FLAGS.
1927-192D	SUB-ID NUMBERS TO BE MATCHED.
192E-192F	LOCATION OF BYTE1 LOW.
1930-1931	LOCATION OF BYTE2 HIGH.
1932	ANALOG1 FLAGS
1933-1939	SUB-ID NUMBERS TO BE MATCHED.
193A-193B	LOCATION OF BYTE1.
193C-193D	SPARE.
193E	ANALOG2 FLAGS.
193F-1945	SUB-ID NUMBERS TO BE MATCHED.
1946-1947	LOCATION OF BYTE.
1948-1949	SPARE

194A	ANALOG3 FLAGS.
194B-1951	SUB-ID NUMBERS TO BE MATCHED.
1952-1953	LOCATION OF BYTE.
1954-1955	SPARE.
1956	ANALOG4 FLAGS.
1957-195D	SUB-ID NUMBERS TO BE MATCHED.
195E-195F	LOCATION OF BYTE.
1960-1961	SPARE.
1962	ANALOG 5 FLAGS.
1963-1969	SUB-ID NUMBERS TO BE MATCHED.
196A-196D	SPARE.
196E-19AD	TM BUFFER FOR SORTING.
19AE-19AF	TTY BYTE LOCATION.
19B0-1A12	TRANSMITION TO BALLOON DATA
1A13-1BFF	STACK.
1C00-27FF	USERS RAM.

2000-27FF IS INDIRECTLY ACCESSIBLE.

FLAGS

18E3 BIT7 (MSB) IS "BUFFER #1 FULL" FLAG.
BIT6 (NSB) IS "BUFFER #2 FULL" FLAG.

WHEN THE "INT7.5" ROUTINE FILLS A BUFFER IT SETS THE APPROPRIATE "BUFFER FULL" FLAG. THIS TELLS OTHER ROUTINES THAT NO MORE DATA WILL BE ENTERED UNTIL THE NEXT BUFFER IS FILLED.

THE "LDBUFF" ROUTINE WILL RESET A "BUFFER FULL" FLAG WHEN IT IS GOINT TO TRANSFER DATA FROM A BUFFER TO A NEW BUFFER FOR PROCESSING. ALSO, THE "INT7.5" ROUTINE CAN RESET "BUFFER FULL" FLAGS IF THEY ARE NOT USED BEFORE IT HAS FILLED THE OTHER BUFFER.

18EA BIT7 (MSB) IS "NO ASCII TO SYSTEM BINARY" FLAG.

IF "TERMIN" IS USED AND THIS FLAG IS SET, THEN THE DATA READ FROM THE TERMINAL WILL NOT BE CONVERTED INTO SYSTEM BINARY.

18EB ANYBIT IS "HIGHYTE LOCATION ONLY" FLAG.

IF "GETALL" IS USED AND THIS FLAG IS SET, THEN ONLY THE "HBYT LOC" QUESTION IS ASKED. USED FOR GETTING LOCATION FO DATA FOR ANALOG CHANNELS (ONLY NEEDS ONE BYTE).

1800 through 180F
BIT 3 IS "DECIMAL" FLAG.
BIT 2 IS "BINARY" FLAG.
BIT 1 IS "INDIRECT" FLAG.
BIT Ø (LSB) IS "DIRECT" FLAG.

DEFINES MODE OF INPUT/OUTPUT PRESENTATION. SEE "DCML", "BNRY," INDRCT", "DRCT" ROUTINE DESCRIPTIONS FOR DETAILS.

1800 - CONTAINS "DISPL" FLAG.
1801 - CONTAINS "ALTER" FLAGS.
1802 - CONTAINS "FILL" FLAGS.
1803 - CONTAINS "MOV" FLAGS.
1804 - SPARE.
1805 - "COMPD","COMPA" FLAGS.
1806 - SPARE.
1807 - CONTAINS "GO" FLAGS.
1808 - SPARE.
1809 - CONTAINS FLAG USED IN DECOMUTATION
REST ARE SPARES

1848 - BIT 7 (MSB) IS "ADDRESS FLAG"

IF ADDRESS IS USED, BIT 7 IS RESET, AND THE "INDIRECT" FLAG OF THE CALLING PROGRAM IS SET, THEN THE OFFSET 2000 WILL NOT BE ADDED. USED FOR GETTING A BYTE OF DATA RATHER THAN AN ADDRESS FROM THE INPUT DEVICE.

1849 - BIT 7 (MSB) IS "RETURN ON CK" FLAG.

IF "GETDAT" IS USED AND THIS FLAG IS SET, AND THE FIRST ENTRY BY THE INPUT DEVICE IS A "CR", THEN "GETDAT" WILL RETURN TO THE CALLING ROUTINE. ALSO RESETS "RETURN ON CR" BEFORE RETURNING.

184D - BIT 7 (MSB) IS "CLEAR DISPLAY" FLAGS.

IF "READ" IS USED AND THIS FLAG IS SET, THEN THE OUTPUT DEVICE WILL BE CLEARED BEFORE RETURNING TO CALLING PROGRAM.

1829 - BIT 7 (MSB) IS "USART/TM" FLAG.

IF "TTYLNK" IS USED AND THIS FLAG IS SET, THEN THE BALLOON DOWN LINK IS DONE BY USART. IF THIS FLAG IS RESET THEN THE BALLOON DOWN LINK IS DONE BY TM.

1855 - BIT 7 (MSB) IS "TERM/GCU" FLAG.

IF THIS FLAG IS SET THEN ALL COMMUNICATIONS TO THE USER ARE DONE BY A TERMINAL. IF THIS FLAG IS RESET THEN ALL COMMUNICATION TO THE USER ARE DONE BY THE CONSOLE.

```
        $TITLE   "BBIMSC WRITTEN BY JIM MARLEY      [illegible]"
        SECTION BBIMC
        GLOBAL  M41
        GLOBAL  M42
        GLOBAL  M22
        GLOBAL  M34
        GLOBAL  M35
        GLOBAL  M36
        GLOBAL  M37
        GLOBAL  MODE
        GLOBAL  BIASP
        GLOBAL  BIASS
        GLOBAL  READ
        GLOBAL  ENDIT
        GLOBAL  GETLAT
        GLOBAL  M3C
        GLOBAL  FRAME
        GLOBAL  NUMLK
        GLOBAL  M14
        GLOBAL  MESSAC
        GLOBAL  FILL
        GLOBAL  MANY
        GLOBAL  LOOKUP
        GLOBAL  CLEAR
RST0    MVI     A,0C0H
        SIM
        JMP     BEGIN
        WORD    0FFFFH
RST1    PUSH    H           ;THIS SUB UPDATES HEX DISPLAY
        PUSH    D
        PUSH    B
        PUSH    PSW
        JMP     RST1A
        BYTE    0FFH
RST2    BYTE    0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
RST3    BYTE    0,0,0,0,0,0,0,0
RST4    BYTE    0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
RST5    BYTE    0FFH,0FFH,0FFH,0FFH
INT55   BYTE    0,0,0,0
RST6    BYTE    0FFH,0FFH,0FFH,0FFH
INT65   JMP     G26
        BYTE    0FFH
RST7    BYTE    0FFH,0FFH,0FFH,0FFH
INT75   PUSH    PSW
        IN      0A0H
        OUT     0A1H
        PUSH    H           ;THIS INTERUPT IS USED TO COLLECT [illegible]
                           DATA BYTES AND STORE IN SWITCHING [illegible]
        PUSH    B           ;SAVE REGISTERS
        PUSH    PSW
        MVI     A,03H       ;SET MONITOR STROBE
        OUT     0A3H
```

```
         LHLD    1FE0H    ;GET BUFFER POINTER
         MOV     A,L      ;FIND OUT HOW MANY BYTES LEFT TO GO
         SUI     60H
         ANI     7FH
         MOV     D,A      ;STORE BYTE NUMBER IN -D-
         LDA     1FE2H    ;GET NUMBER OF BYTES IN PHASE
         CMP     D        ;IS THIS THE LAST BYTE TO COLLECT?
         JZ      G5       ;JUMP IF IT IS
G6       POP     PSW      ;GET ENTERED DATA AND STORE IT IN
                                                           BUFFER
         MOV     M,A
         INR     L        ;POINT TO NEXT LOCATION AND STORE IT
         SHLD    1FE0H
         MVI     A,02H    ;TURN OFF MONITOR STROBE
         OUT     0A3H
         POP     B
         POP     H
         POP     PSW
         EI
         RET
G5       MOV     A,L      ;CHANGE BUFFERS
         CPI     0ACH     ;IS IT THE FIRST BUFFER
         JC      G6       ;JUMP IF IT IS AND SWITCH TO SECOND
                                                           BUFFER
         MVI     L,60H    ;IF SECOND SWITCH TO FIRST
         MVI     A,40H    ;SET BUFFER FULL FLAG
G7       STA     1FE3H    ;STORE FLAGS
         JMP     G8
G6       MVI     L,0ACH
         MVI     A,80H
         JMP     G7
S20      CALL    CLEAR    ;ENTRY WHEN IN MAIN USED TO ESCAPE
         XRA     A
         OUT     0F0H
         OUT     0F1H
         OUT     0F2H
         OUT     0F3H
         OUT     0F4H
         OUT     0F5H
         OUT     0F6H
         OUT     0F7H
         OUT     0F8H
         OUT     0F9H
         IN      0A2H
         ORI     0F1H
         OUT     0A2H
         IN      90H
         JNZ     C9BACK
DIGI5    LXI     H,C9BACK
         SHLD    182AH
         XRA     A
         STA     1FE2H    ;RESET PHASE LENGTH
```

```
STA     1FE7H   ;RESET BUFFER FLAG
STA     1981H   ;RESET ALL F  FLAGS
STA     1929H   ;COMMUNICATION PORT OVER TO LINE
LXI     H,1850H
SHLD    1800H
XRA     A       ;RESET TERMINAL FLAG CONSOLE IN CONTROL
STA     1855H
MVI     A,0CEH  ;SET UP USART
OUT     91H
OUT     0C1H
MVI     A,37H   ;RESET ANY ERRORS AND ENABLE TRANS AND
                                                  RECV
OUT     91H
OUT     0C3H
MVI     A,7EH   ;TURN ON THE USART
OUT     0D3H
MVI     A,40H
OUT     0D1H
MVI     A,01H   ;CLOCK SET FOR 300 BAUD
OUT     0D1H
MVI     A,3EH
OUT     0D3H
MVI     A,0A0H
OUT     0D0H
MVI     A,00H
OUT     0D0H
MVI     A,0FFH
STA     1848H
STA     1852H
LXI     SP,1BFFH;SET UP THE STACK POINTER
MVI     A,05H
LXI     H,1800H
LXI     D,180FH
CALL    FILL
MVI     A,80H   ;NOW SET ONE 8255A TO ALL OUTPUT
OUT     0B3H
MVI     A,90H   ;SET THE OTHER TO TWO OUTPUTS ONE INPUT
OUT     0A3H
MVI     A,0FEH  ;TURN OFF ALL BUT CONSOLE LEDS
OUT     0A2H
MVI     A,08H   ;SET 8279A TO 16 DIGIT DISPLAY 2 KEY
                                                  ROLLOVER
OUT     C1H
CALL    CLEAR
LXI     H,1950H
SHLD    1FE0H
LXI     H,1900H
LXI     D,191FH
MVI     A,0B0H
CALL    FILL
LXI     H,0C000H
LXI     D,0FFFFH
```

```
MVI     A,00H
CALL    FILL
MVI     A,10H
OUT     0A2H
LXI     H,M14     ;DISPLAY ...C-2 MESSAGE
MVI     A,05H
CALL    MESSAG
LXI     B,0003H
LXI     D,0008H
LXI     H,1900H
MVI     M,00H
DAD     D
MVI     M,78H
INX     H
MVI     M,19H
INX     H
MVI     M,7AH
INX     H
MVI     M,19H
DAD     D
MVI     M,00H
DAD     D
MVI     M,77H
INX     H
MVI     M,19H
INX     H
MVI     M,76H
INX     H
MVI     M,19H
INX     H
MVI     M,00H
DAD     D
MVI     M,73H
INX     H
MVI     M,19H
INX     H
MVI     M,72H
INX     H
MVI     M,19H
INX     H
MVI     M,80H
DAD     D
MVI     M,70H
INX     H
MVI     M,19H
INX     H
MVI     M,7CH
INX     H
MVI     M,19H
INX     H
MVI     M,91H
INX     H
```

```
    MVI     M,00H
    INX     H
    MVI     M,04H
    INX     H
    MVI     M,08H
    INX     H
    MVI     M,0CH
    INX     H
    MVI     M,10H
DAD     D
    MVI     M,71H
    INX     H
    MVI     M,19H
DAD     D
    MVI     M,91H
    INX     H
    MVI     M,64H
DCX     D
DAD     D
    MVI     M,71H
    INX     H
    MVI     M,19H
DAD     D
    MVI     M,91H
    INX     H
    MVI     M,0CH
DAD     D
    MVI     M,71H
    INX     H
    MVI     M,19H
DAD     D
    MVI     M,91H
    INX     H
    MVI     M,10H
    INX     H
    MVI     M,04H
    INX     H
    MVI     M,08H
    INX     H
    MVI     M,10H
    INX     H
DAD     D
    MVI     M,71H
    INX     H
    MVI     M,19H
DAD     D
    MVI     M,91H
    INX     H
    MVI     M,00H
DAD     D
    MVI     M,71H
    INX     H
```

```
         MVI    M,19H
         LXI    H,19[illegible]H
         MVI    M,79H
         INX    H
         MVI    M,19H
         LXI    H,1[illegible]AH
         MVI    M,7AH
         INX    H
         MVI    M,19H
         INX    H
         MVI    M,78H
         INX    H
         MVI    M,19H
         INX    H
         MVI    M,7FH
         INX    H
         MVI    M,19H
G999     MVI    A,0FFH
         STA    1CEAH
         CALL   MANY
         MOV    A,B       ;WAS THERE ANY ENTRY IN SUB MANY?
         CPI    00H
         JZ     G999      ;JUMP IF THERE WASNT ANY ENTRIES
         XRA    A
         STA    1CEAH
         CALL   LOOKUP
         JC     G999      ;JUMP IF ERROR IN SUB LOOKUP
         PUSH   B         ;SET UP SO WE CAN RETURN
         LXI    B,CMBACK
         XTHL
         PCHL
CMBACK   LXI    SP,1CCCH
         MVI    A,0FFH    ;SET ADDRESS STATUS
         STA    1C48H
         JMP    G999
NSPLD    MVI    E,06H     ;-BE- CONTAINS HEX CHARECTOR
         MVI    A,90H     ;DISPLAY THEM ONE AT A TIME IN AUTO
                                                     INDEX MODE
         OUT    31H
         LXI    B,1C1[illegible]H ;-BC- CONTAINS LOCATION OF DATA IN
                                                         MEMORY
G7       LXI    H,NUMBER  ;-HL- CONTAINS LOOKUP LOCATION
         LDAX   B         ;GET A CHERECTOR
         CPI    0A0H      ;IS IT A SPACE?
         JNZ    G3        ;JUMP IF IT ISNT
         MVI    A,0FFH    ;SET ALL SEGMENTS
         JMP    G4
G3       MOV    L,A       ;PUT CHARECTOR INTO -HL-
         DAD    D         ;ADD TO POINTER
         MOV    A,M       ;GET SEGMENT PATTERN
G4       OUT    30H       ;OUTPUT IT
         MOV    A,C       ;IS THIS THE LAST CHARECTOR
```

```
        INX     B           ;POINT T  NEXT IF IT ISNT
        CPI     27H
        JNZ     G7          ;JUMP BACK IF MORE CHARACTERS
        POP     PSW
        POP     B
        POP     D
        POP     H
        RET
FRAME   MVI     C,00H       ;THIS SUB GETS US INTO SYNC
        MVI     A,0DH
        SIM
        EI
G9      IN      0ACH        ;INPUT TM DATA UNTIL YOU GET EB
                                                        SYNC
        CPI     0EBH
        JNZ     G9          ;JUMP IF NOT EB
        MVI     A,05H       ;SYNCRONIZE WITH CLOCK
        OUT     0A3H
        MVI     A,04H
        OUT     0A3H
        MVI     A,18H       ;RESET INT75 FLIP FLOP FLAG
        SIM
G10     RIM                 ;WAIT FOR INT75 FLIP FLOP FLAG TO BE
                                                        SET
        ANI     40H
        JZ      G10         ;JUMP IF ITS NOT SET
        IN      0ACH        ;GET NEXT TM BYTE
        CPI     90H         ;IS IT THE SECOND HALF OF SYNC?
        JNZ     G9          ;JUMP IF ITS NOT
G11     MVI     B,00H       ;SET LOOP COUNTER TO ZERO
G12     MVI     A,10H       ;RESET INT75 FLIP FLOP
        SIM
G13     RIM                 ;WAIT FOR INT75 FLIP FLOP TO BE SET
        ANI     40H
        JZ      G13         ;JUMP IF ITS NOT
        IN      0ACH        ;GET NEXT TM BYTE
        CPI     90H         ;IS IT FIRST SYNC BYTE
        JNZ     G14         ;THIS WILL COUNT THE NUMBER OF BYTES
                                                     PER FRAME
        MOV     A,C         ;IS THIS SECOND LOOP?
        RRC
        JC      G15         ;JUMP IF IT IS
        MOV     H,B         ;SAVE NUMBER OF BYTES FOR COMPARISON
                                                     LATER
        INR     C           ;INCREAMENT LOOP COUNTER
        JMP     G11
G14     INR     B           ;INCREAMENT BYTE COUNTER
        JMP     G12
G15     MOV     A,H         ;GET FIRST LOOP COUNT OF BYTES PER
                                                     FRAME
        CMP     B           ;IS IT EQUAL TO SECOND LOOPS COUNT?
        JNZ     FRAME       ;IF ITS NOT GO FRAME AGAIN
```

```
        INR     A         ;CORRECT NUMBER OF BYTES TO FLUSH ONE
        CPI     70H       ;IF FRAME IS TO LONG LOAD WITH 3F
        JC      G16
        MVI     A,3FH
G16     STA     1E77H     ;STORE FRAME LENGTH
        MVI     A,19H     ;TURN ON INT75 AND INT65 AND ENABLE
                                                        INTERRUPTS
        SIM
        EI
        RET
YORN    CALL    READ      ;THIS SUB GETS A YES OR NO ANSWER
        CPI     9BH       ;IS ENTRY AN ESC?
        JZ      G17       ;JUMP IF IT IS
        CPI     0CEH      ;IS IT A NO?
        STC
        RZ                ;RETURN WITH CY SET IF IT IS
        CPI     0D9H      ;IS IT A YES
        JNZ     YORN      ;JUMP IF ITS NOT A YES OR NO
        RET
G17     CALL    EMLIT
        LHLD    1E2AH
        PCHL
MODE    LXI     H,M33     ;THIS SUB DEFINES MODE USED FOR MEAS.
        MVI     A,05H
        CALL    MESSAG    ;DISPLAY DOWN? MESSAGE
        CALL    YORN      ;GET ANSWER
        MVI     B,80H     ;SET NO DOWN BIT AND JUMP IF THIS IS
                                                          CORRECT
        JC      G18
        MVI     B,00H     ;RESET NO DOWN FLAG
G18     LXI     H,M34
        MVI     A,06H
        CALL    MESSAG    ;DISPLAY AMU SWP? MESSAGE
        CALL    YORN      ;GET ANSWER
        MOV     A,B       ;GET FLAGS
        JC      G19       ;JUMP AND SET BIAS SWEEP FLAG
        ORI     20H       ;SET AMU SWEEP FLAG
G20     MOV     B,A       ;SAVE FLAGS
        LXI     H,M35
        MVI     A,08H
        CALL    MESSAG    ;DISPLAY TOTIONS? MESSAGE
        CALL    YORN      ;GET ANSWER
        JC      G21       ;RESET TOTAL IONS FLAG
        MOV     A,B
        ORI     10H       ;SET TOTAL IONS FLAG
        MOV     B,A
G21     LXI     H,M36
        MVI     A,07H
        CALL    MESSAG    ;DISPLAY ACCUM? MESSAGE
        CALL    YORN      ;GET ANSWER
        JC      G22       ;RESET ACCUMULATION FLAG
        MOV     A,B
```

```
        ORI     80H     ;SET ACCUMULATION FLAG
        MOV     B,A
        LXI     H,M27
        MVI     A,07H
        CALL    MESSAG  ;DISPLAY SWITCH? MESSAGE
        CALL    YORN    ;GET ANSWER
        MOV     A,B
        JC      G27     ;RESET SWITCH FLAG
        ORI     40H     ;SET SWITCH FLAG
G27     STA     2409H   ;SET UP TO CONVERT AND STORE
        LXI     H,M41   ;DISPLAY STEPING? MESSAGE
        MVI     A,0CH
        CALL    MESSAG
        CALL    YORN
        JC      G31
G20     LXI     H,M42   ;DISPLAY VALUE MESSAGE
        MVI     A,05H
        CALL    MESSAG
        LXI     H,1809H
        CALL    GETDAT
        JC      G29
G30     STA     2415H
        CALL    ENDIT
        RET
G31     MVI     A,00H
        JMP     G30
G19     ORI     40H
        JMP     G20
BIAS1   LXI     D,240AH ;THIS SUB DEFINES PRIMARY BIASES
                                                        WITH
        JMP     G24
BIAS2   LXI     D,240FH ;THIS SUB DEFINES SECONDARY BIASES
                                                        WITH
G24     MVI     B,01H   ;SET BIAS IDENTIFIER TO
        PUSH    D
G25     LXI     H,M38
        MVI     A,04H
        CALL    MESSAG  ;DISPLAY BIAS MESSAGE
        MOV     A,B     ;GET BIAS IDENTIFIER AND DISPLAY IT
        STA     3805H
        PUSH    B
        LXI     H,1809H
        MVI     A,0FFH
        STA     1849H
        CALL    GETDAT  ;GET ANSWER
        JC      G1      ;JUMP IF NOT
        POP     B
        XCHG
        MOV     M,A
        INX     H
        XCHG
G27     INR     B       ;POINT TO NEXT BIAS
```

```
         STITLE  "BBIMS1 WRITTEN BY JIM MANLEY      SECOND OF
                                                         SIX"
         GLOBAL  LOOKP
         GLOBAL  M1
         GLOBAL  M2
         GLOBAL  M3
         GLOBAL  M4
         GLOBAL  M8
         GLOBAL  M9
         GLOBAL  M10
         GLOBAL  M12
         GLOBAL  MOVE
         GLOBAL  TRMOUT
         GLOBAL  G5B2
         GLOBAL  TERMIN
         GLOBAL  ERROR
         GLOBAL  READ
         GLOBAL  FILL
         GLOBAL  DIRECT
         GLOBAL  INDRCT
         GLOBAL  BNRY
         GLOBAL  DCML
         GLOBAL  MOVEM
         GLOBAL  MANY
         GLOBAL  LOOKUP
         GLOBAL  CLEAR
         GLOBAL  GETDAT
         GLOBAL  DECBIN
         GLOBAL  BINDEC
         GLOBAL  ADDRES
         GLOBAL  BELL
         GLOBAL  MESSAG
         GLOBAL  NMREAD
         GLOBAL  CMPDH
         GLOBAL  ENDIT
         GLOBAL  ALREAD
READ     LDA     1855H   ;THIS SUB GETS DATA FROM KEYBRD OR
                                                    TERMINAL
         ORA     A
         JM      TERMIN  ;ENTER DATA THROUGH TERMINAL IF FLAG IS
                                                         SET
         IN      81H     ;CHECK KEYBOARD FOR ENTRY
         ORA     A
         JM      READ    ;IF MINUS THEN KEYBOARD IS BUSY
         ANI     0FH     ;IS THERE ANY DATA?
         CPI     00H
         JZ      READ    ;JUMP BACK IF NO DATA AND WAIT
         MVI     A,40H   ;TELL KEYBOARD YOU ARE ABOUT TO READ IT
         OUT     81H
         IN      80H
         ANI     3FH     ;AFTER RESPONSE MASK WITH 3F
         CPI     10H     ;IS IT A NUMBER?
```

```
        JC      G582     ;JUMP IF IT IS
        JNZ     G59      ;JUMP IF ITS NOT A 10
        MVI     A,8DH    ;LOAD -ACC- WITH ASCII CR
G59     CPI     11H      ;IS ENTRY AN 11
        JNZ     G8       ;JUMP IF ITS NOT AN 11
        MVI     A,9BH    ;LOAD -ACC- WITH ASCII ESC
G8      CPI     18H      ;IS IT A GO?
        JNZ     G68      ;JUMP IF ITS NOT
        MVI     A,91H
G68     CPI     31H      ;IS IT A YES ENTRY?
        JNZ     G66      ;JUMP IF ITS NOT
        MVI     A,0D9H   ;LOAD -ACC- WITH ASCII Y
G66     CPI     32H      ;IS IT A NO ENTRY?
        JNZ     G67      ;JUMP IF ITS NOT
        MVI     A,0CEH   ;LOAD -ACC- WITH ASCII N
G67     ORI     80H
        CPI     0B0H     ;IS IT A BACK SPACE?
        JNZ     G582     ;JUMP IF ITS NOT
        MVI     A,88H
G582    PUSH    PSW      ;SAVE IT
        LDA     184DH    ;CHECK TO SEE IF WE HAVE TO CLEAR
                                                         DISPLAY
        ORA     A
        JP      G80      ;CLEAR IF MINUS
        CALL    CLEAR    ;CALL CLEAR IF CONSOLE IS OPERATABLE
        MVI     A,8DH    ;OUTPUT TO TERMINAL A CR LF
        CALL    TRMOUT
        MVI     A,8AH
        CALL    TRMOUT
G9      XRA     A        ;RESET THE CLEAR FLAG
        STA     184DH
G80     PUSH    H        ;SAVE -HL-
        LXI     H,04FFH  ;SET UP TO RING BELL
        CALL    BELL
        POP     H
        POP     PSW
        RET
NMREAD  PUSH    H        ;THIS SUB READS IN ONLY NUMBERS FROM
                                                     THE KEYBRD
        PUSH    D
        PUSH    B
G1      CALL    READ     ;GO GET AN ENTRY
        MOV     B,A      ;SAVE ENTRY
        CPI     9BH      ;IS IT AN ESCAPE REQUEST?
        JZ      G2       ;JUMP IF IT IS
        CPI     88H      ;IS IT A BACK SPACE REQUEST?
        JZ      G5       ;JUMP IF IT IS
        CPI     8DH      ;IS IT A CR?
        JZ      G3       ;JUMP IF IT IS
        CPI     10H      ;IS IT A NUMBER?
        JNC     G6       ;JUMP IF IT ISNT
        LXI     H,181CH  ;SET UP TO SHIFT NUMBER DISPLAY LEFT
```

```
                                                                   ONE
        LXI     D,181DH ;DE CONTAINS HL-1,HL CONTAINS MSDIGIT
                                                            OF DISPLAY
G4      MOV     A,M     ;START SHIFTING EACH ONE DOWN ONE
                                                              LOCATION
        STAX    D
        MOV     A,L
        DCX     H
        DCX     D
        CPI     18H     ;IS THIS THE LAST SHIFT?
        JNZ     G4      ;JUMP IF IT ISNT
        MOV     A,B
        STAX    D
        RST     1       ;DISPLAY ALL
G3      POP     B
        POP     D
        POP     H
        RET
G6      LXI     H,3FFFH ;SET UP TO RING BELL
        CALL    BELL
        JMP     G1      ;TRY FOR CORRECT ENTRY
G2      CALL    CLEAR   ;CLEAR DISPLAY AND GO TO ESCAPE
                                                              LOCATION
        LHLD    182AH   ;GET ESCAPE DATA AND PUT IN HL SO WE
                                                                   CAN
        PCHL            ;REPLACE PROGRAM COUNTER WITH ESCAPE
                                                                  DATA
G5      LXI     H,1819H ;SET UP TO SHIFT NUMBER DISPLAY TO THE
                                                             RIGHT ONE
        LXI     D,1818H
G70     MOV     A,M
        STAX    D
        INX     H
        INX     D
        MOV     A,L
        CPI     1DH
        JNZ     G70
        MVI     A,0A0H
        STA     181DH
        RST     1       ;DISPLAY ALL
        JMP     G1      ;GO GET ANOTHER NUMBER
MOVE    MOV     A,M     ;SUB TO MOVE DATA FROM HL TO DE TO NEW
                                                   LOCATION BC AND UP
        STAX    B
        CALL    CMPDH   ;SEE IF THIS IS THE END
        RZ              ;RETURN IF IT IS
        INX     H       ;INC HL AND BC , POINT TO NEXT MOVE
                                                              LOCATION
        INX     B
        JMP     MOVE    ;GO MOVE THE NEXT BYTE
CMPDH   MOV     A,D     ;THIS SUB COMPARES HL TO DE
        CMP     H       ;IF HL>DE THEN CY=1
```

```
        RNZ
        MOV     A,E       ;HL=DE THEN Z=1,CY=0
        CMP     L         ;IF HL<DE THEN CY=0
        RET
ALREAD  PUSH    H         ;SUB READS IN ALFA-NUMERIC CHARECTORS
                                    AND LOADS IN TO BE DISPLAYED
        PUSH    D
        PUSH    B
G11     CALL    READ      ;GET A CHARECTOR
        MOV     B,A       ;SAVE
        CPI     88H       ;IS A BACK SPACE REQUEST?
        JZ      G12       ;JUMP IF IT IS
        CPI     9BH       ;IS IT AN ESCAPE REQUEST?
        JZ      G13       ;JUMP IF IT IS
        CPI     8DH       ;IS IT A CR?
        JZ      G14       ;JUMP IF IT IS
        LXI     D,1817H   ;SET UP TO SHIFT ALFA TO THE LEFT BY
                                                            ONE
        LXI     H,1816H
G15     MOV     A,M       ;GET DATA AND MOVE TO THE LEFT
        STAX    D
        DCX     H         ;POINT TO NEXT LOCATION TO MOVE
        DCX     D
        MOV     A,L       ;SEE IF THIS IS THE LAST LOCATION TO
                                                           MOVE
        CPI     0FH
        JNZ     G15       ;JUMP IF MORE MOVES ARE NEEDED
        MOV     A,B       ;GET BACK SAVED DATA
        STAX    D         ;STORE NEW DATA IN LSDIGIT
G14     POP     B
        POP     D
        POP     H
        RET
G12     LXI     H,1811H   ;SHIFT ALFA STORAGE TO THE RIGHT ONE
        LXI     D,1817H
        LXI     B,1810H
        CALL    MOVE
        MVI     A,0A0H
        STA     1817H
        JMP     G11
G13     LHLD    182AH     ;LOAD ESCAPE DATA INTO HL SO IT CAN BE
                                                PUT INTO THE PC
        CALL    CLEAR     ;CLEAR THE DISPLAYS
        PCHL              ;HL MOVED INTO PC
BELL    PUSH    D         ;SUB RINGS THE BELL FOR (HL) LONG
        PUSH    PSW
        LXI     D,0FFFEH
        MVI     A,06H     ;SET UP TO START RINGING BELL
        OUT     0B3H
G16     DAD     D
        JC      G16       ;CONTINUE IF NOT DONE
        MVI     A,07H     ;SET UP TO STOP BELL
```

```
        OUT     0B3H
        POP     PSW
        POP     D
        RET
DIRECT  LDA     1810H    ;THIS SUB SETS THE DIRECT ACCESS FLAG
                                          OF OTHER PROGRAMS
        CPI     0BDH     ;WAS THE LAST KEYBRD ENTRY AN = ?
        JNZ     G20      ;JUMP IF IT WASNT
G17     CALL    MANY     ;LOOKUP THE PROGRAMS YOU WANT TO BE
                                      DIRECT,IF THE PROGRAM
        MOV     A,B      ;WERE THERE ANY ENTRIES IN SUB MANY?
        CPI     00H
        JZ      G17      ;JUMP IF THERE WERENT ANY ENTRIES
        CALL    LOOKUP   ;IS DIRECT THEN IT IS ALSO BINARY
        JC      G18      ;JUMP IF THERE IS AN ERROR
        LXI     H,8000H  ;LETS MAKE SURE THIS PROGRAM IS ABLE TO
                                              ACCEPT DIRECT
        CALL    CMPDH
        JNC     G18      ;JUMP IF THE PROGRAM IS NOT ABLE TO
                                    ACCEPT DIRECT COMMANDS
        LDAX    D        ;GET LOOKED UP PROGRAMS STATUS
        ANI     0F5H     ;MASK OUT INDIRECT AND DECIMAL FLAGS
        ORI     05H      ;MASK IN DIRECT AND BINARY FLAGS
        STAX    D        ;STORE FLAGS IN MEMORY
        LDA     1810H    ;LOOK AT LAST KEYBRD ENTRY
        CPI     0BDH     ;IS IT AN = ?
        JZ      G17      ;JUMP IF IT IS AND GET ANOTHER PROGRAM
                                                  TO UPDATE
        JMP     G19      ;END IF IT WASNT AN =
G18     CALL    ERROR    ;DISPLAY ERROR MESSAGE AND GO GET
                                 ANOTHER PROGRAM TO UPDATE
        JMP     G17
G20     LXI     D,180FH  ;IN THIS MODE ALL PROGRAMS ARE SET TO
                                         DIRECT AND BINARY
        LXI     H,17FFH
G21     INX     H        ;INCREMENT HL TO THE NEXT MEMORY
                                                   LOCATION
        MOV     A,M      ;GET PROGRAM STATUS FLAGS
        ANI     0F5H     ;MASK OUT DECIMAL AND INDIRECT FLAGS
        ORI     05H      ;MASK IN BINARY AND DIRECT FLAGS
        MOV     M,A      ;STORE STATUS
        CALL    CMPDH    ;IS THIS THE LAST PROGRAM TO BE
                                                   UPDATED?
        JNZ     G21      ;JUMP IF IT WASNT THE LAST ONE
G19     CALL    ENDIT    ;END THIS PROGRAM
        RET
ERROR   CALL    CLEAR    ;THIS SUB DISPLAYS ERROR MESSAGE CLEARS
                                                HEX DISPLAY
        MVI     A,05H    ;SET UP TO DISPLAY ERROR MESSAGE
        LXI     H,M9
        CALL    MESSAG
        LXI     H,3FFFH  ;SET UP TO RING BELL
```

```
        CALL    BELL
        RET
ENDIT   MVI     A,03H   ;THIS SUB DISPLAYS END MESSAGE AND
                                             CLEARS HEX DISPLAY
        LXI     H,M10   ;SET UP TO DISPLAY END MESSAGE
        CALL    MESSAG
        RET
MESSAG  PUSH    D       ;THIS SUB DISPLAYS A MESSAGE OF LENGTH
                                                -A- AND LOCATED
        PUSH    B       ;AT -HL- AND UP
        CALL    CLEAR   ;SET UP TO LOAD DISPLAY WITH SPACES
        XCHG            ;MOVE -HL- INTO -DE-
        MOV     L,A     ;PUT NUMBER OF CHARECTORS INTO -HL-
        MVI     H,00H
        DCR     L
        DAD     D       ;HL+DE
        LXI     B,3800H ;ADDRESS IN -DE- AND AN END ADDRESS IN
                                                   -HL- ALSO A
        XCHG
G78     MOV     A,M     ;NOW MOVE MESSAGE TO DISPLAY MEMORY
        ORI     80H     ;SET MSBIT TO SHOW SYSTEM BINARY
        STAX    B
        CALL    TRMOUT
        CALL    CMPDH
        INX     B
        INX     H
        JNZ     G78
        POP     B
        POP     D
        MVI     A,8DH   ;SET UP TO TRANSMIT CR LF
        CALL    TRMOUT
        MVI     A,8AH
        CALL    TRMOUT
        RET
FILL    MOV     M,A     ;MOVE THE -ACC- INTO MEMORY UNTIL -HL-
                                                        = -DE-
        STA     184CH
        CALL    CMPDH
        RZ              ;RETURN IF THEY ARE EQUAL
        LDA     184CH
        INX     H       ;POINT TO NEXT MEMORY LOCATION
        JMP     FILL    ;JUMP BACK AND CONTINUE
CLEARB  PUSH    PSW        ;THIS SUB CLEARS ONLY HEX DISPLAY
        MVI     A,0DCH
        OUT     81H
        POP     PSW
        RET
CLEAR   PUSH    H       ;OUTPUT A CLEAR INSTRUCTION TO THE
                                                      8279A-5
        PUSH    D
        PUSH    PSW
        MVI     A,0DFH
```

```
        OUT     81H
G22     IN      81H     ;CHECK MSBIT OF STATUS FROM 8279A-5
        ORA     A
        JM      G22     ;IF MINUS THEN STAY IN LOOP
        LXI     H,3800H
        LXI     D,3807H
        MVI     A,0A0H
        CALL    FILL
        LXI     H,1810H ;NOW CLEAR DISPLAY MEMORY
G64     LXI     D,1827H
        MVI     A,0A0H
        CALL    FILL
        MVI     A,8DH   ;OUTPUT A CR LF
        CALL    TRMOUT
        MVI     A,8AH
        CALL    TRMOUT
        POP     PSW
        POP     D
        POP     H
        RET
MANY    MVI     B,00H   ;THIS SUB GETS KEYBRD ENTRIES AND
                                                DISPLAYS THEM
        LDA     1855H   ;IF TERMINAL IS ACTIVE THEN DONT SET
                                                  CLEAR FLAG
        ORA     A
        JM      G23
        MVI     A,0FFH  ;SET CHARECTOR COUNTER TO 00 THEN GET A
                                                   CHARECTOR
        STA     184DH   ;SET THE CLEAR FLAG FOR SUB READ
G23     CALL    ALREAD
        CPI     8DH     ;IS ENTRY A CR ?
        RZ              ;RETURN IF IT WAS
        INR     B       ;INCREAMENT CHARECTOR COUNTER
        MOV     A,B     ;SEE IF B IS TO LARGE
        CPI     09H
        JNZ     G23     ;JUMP IF IT ISNT
        MVI     B,08H   ;NOT TO EXCEED 8 CHARECTORS
        JMP     G23     ;GO GET NEXT CHARECTOR
LOOKUP  LXI     H,LOOKP ;THIS SUB LOOKS UP PROGRAM STARTING
                                        ADDRESSES AND STATUS
G24     MVI     A,10H   ;ADJUST -DE- SO IT POINTS TO THE FIRST
                                                   CHARECTOR
        DCR     B       ;CORRECT -B- THE CHARECTOR COUNTER
        ADD     B       ;ADD ADJUSTMENT TO CHARECTOR COUNTER
        MOV     E,A     ;MOVE ADJUSTED LOW ORDER ADDRESS TO -E-
        MVI     D,18H   ;ATTACH HIGH ORDER ADDRESS
G76     PUSH    D       ;SAVE ENTRY CHARECTOR POINTER
G25     LDAX    D       ;GET A CHARECTOR TO BE LOOKED UP
        ANI     7FH
        CMP     M       ;IS THE LOOKED UP CHARECTOR EQUAL TO
                                     THE ENTERED CHARECTOR ?
        JZ      G26     ;JUMP IF IT IS
```

```
        CPI     3DH     ;CHECK FOR = SIGN,COULD BE VALID ENTRY
        JNZ     G27
        MVI     A,20H    ;IF LAST COMPARISON WAS TRUE THEN
                              CHECK LOOKUP TABLE FOR A SPACE
        CMP     M
        JNZ     G27     ;IF IT IS A SPACE THEN VALID LOOKUP GO
                                             GET ADDRESSES
        DCX     H       ;CORRECT -HL-
        JMP     G28
G27     MVI     A,20H   ;IF NOT THEN LOOK THROUGH LOOKUP TABLE
                                   UNTIL A SPACE IS FOUND
        CMP     M       ;IS IT A SPACE?
        INX     H
        JNZ     G27     ;JUMP IF IT ISNT
        LXI     D,0004H ;ADD 4 TO -HL- , THIS WILL POINT TO
                                        NEXT PROGRAM LIST
        DAD     D
        POP     D       ;GET BACK -DE-
        LDA     1855H   ;CHECK FOR ACTIVE CONSOLE
        RLC
        JC      G7      ;UMP IF ITS NOT
        MOV     A,E     ;MUST HAVE ONLY ONE ENTRY
        CPI     10H
        JNZ     G10     ;IF MORE THAN ONE THEN ERROR
        LDAX    D       ;GET CHARECTOR
        CMP     M       ;ARE THEY EQUAL?
        JNZ     G7      ;UMP IF THEY ARE NOT
        MVI     D,0FFH  ;SET UP TO GO BACK TO BEGINNING OF THIS
                                                     LIST
        INX     H
        MOV     E,M     ;GET 2S COMPLIMENT NEEDED
        DAD     D
        LXI     D,3800H ;POINT TO ALPHA DISPLAY
G61     MOV     A,M     ;DISPLAY PROGRAM NAME
        CPI     20H     ;IS IT THE SPACE?
        JZ      G60     ;JUMP IF IT IS
        ORI     80H
        STAX    D       ;DISPLAY IT
        INX     D
        INX     H
        JMP    G61
G7      INX     H       ;POINT TO NEXT PROGRAM LIST
        INX     H
        MOV     A,M     ;ARE THERE ANY MORE PROGRAM LISTS ?
        CPI     0FFH
        JNZ     G76     ;JUMP IF THERE ARE
G10     CALL    ERROR   ;IF THERE ARENT ANY MORE PROGRAMS THAN
                                             DISPLAY ERROR
        STC             ;SET ERROR FLAG
        RET
G26     MOV     A,E     ;IS THIS THE LAST ENTRY ?
        CPI     10H
```

```
        JZ      G28       ;JUMP IF IT IS
        INX     H         ;POINT TO NEXT LOOKUP LOCATION AND
                                              ENTRY LOCATION
        DCX     D
        JMP     G25
G28     INX     H         ;THIS IS OUR PROGRAM SO GET THE
                                            STARTING ADDRESS
        MOV     A,M       ;MAKE SURE THIS IS THE END OF A LOOKUP
        CPI     20H
        JNZ     G27
        POP     D
G62     INX     H
        MOV     E,M
        INX     H
        MOV     D,M
        PUSH    D         ;SAVE STARTING ADDRESS
        INX     H         ;NOW GET PROGRAM STATUS
        MOV     E,M
        INX     H
        MOV     D,M
        POP     H         ;GET BACK STARTING ADDRESS
        ORA     A         ;RESET ERROR FLAG
        RET
G60     DCX     H         ;GET LAST LETTER OF PROGRAM
        MOV     A,M
        ORI     80H
        STA     1810H     ;STORE IT
        INX     H
        PUSH    H
        LXI     H,0000H
        LXI     D,0001H
G69     DAD     D
        DAD     D
        DAD     D
        JNC     G69
        POP     H
        JMP     G62
ADDRES  PUSH    H         ;THIS SUB GETS A NUMBER UP TO 4 DIGITS
                                       AND CONVERTS IF NES
        CALL    CLEARB
G29     MVI     B,00H     ;SET CHARECTOR COUNTER TO ZERO
        LXI     H,1818H   ;POINT TO MEMORY LOCATION OF DISPLAY
G30     CALL    NMREAD
        CPI     8DH       ;READ IN A NUMBER AND SEE IF IT IS A CR
        JNZ     G30       ;JUMP IF IT ISNT
G31     MOV     A,M       ;COUNT HOW MANY CHARECTORS WERE ENTERED
        CPI     0A0H      ;IS THIS A SPACE?
        JNZ     G32       ;JUMP IF IT WASNT THE LAST NUMBER
        MOV     A,B       ;WAS THERE TO MANY NUMBERS ENTERED?
        CPI     05H
        JNC     G33       ;JUMP IF THERE ARE TO MANY
        MOV     A,B       ;ADJUST THE MEMORY POINTER TO POINT TO
```

```
                                           THE LAST NUMBER ENTERED
        ADI     17H
        MOV     L,A       ;PUT LOW BYTE OF ADDRESS IN -L-
        MVI     H,18H     ;PUT HIGH BYTE OF ADDRESS I'  -H-
        LXI     D,0000H
        MOV     A,B       ;FIND OUT WERE TO START COMBINING DATA
        CPI     01H       ;ARE THERE 1?
        JC      G99       ;JUMP IF THERE ARENT ANY
        JZ      G37       ;JUMP IF THERE IS 1
        CPI     03H       ;ARE THERE 2 OR 3?
        JC      G36       ;JUMP IF THERE ARE 2
        JZ      G35       ;JUMP IF THERE ARE 3
        JMP     G34       ;JUMP IF THERE ARE 4
G99     STC               ;SET THE CARRY TO SHOW ERROR
        POP     H
        RET
G32     MOV     A,L
        CPI     2DH       ;IS THIS THE LAST DISPLAY LOCATION?
        JZ      G33       ;JUMP IF IT IS
        INR     B         ;INCREAMENT NUMBER COUNTER
        INX     H         ;POINT TO NEXT MEMORY LOCATION
        JMP     G31
G33     CALL    CLEARB    ;CLEAR THE DISPLAY AND RING THE BELL
        LXI     H,3FFFH
        CALL    BELL
        JMP     G29       ;JUMP BACK AND START OVER AGAIN
G34     MOV     A,M       ;IF THERE ARE 4 THAN START COMPACTING
                                                          HERE
        RLC
        RLC               ;MAKE THIS NIBBLE HIGH ORDER
        RLC
        RLC
        MOV     D,A       ;SAVE IN D
        DCX     H         ;POINT TO NEXT LOCATION
G35     MOV     A,M       ;IF THERE ARE 3 THAN START HERE
        ORA     D         ;COMBINE -A- AND -D-
        MOV     D,A       ;STORE IN D
        DCX     H         ;POINT TO NEXT LOCATION
G36     MOV     A,M       ;IF THERE ARE 2 THAN START HERE
        RLC
        RLC               ;MAKE THIS NIBBBLE HIGH ORDER
        RLC
        RLC
        MOV     E,A       ;SAVE IN -E-
        DCX     H         ;POINT TO NEXT LOCATION
G37     MOV     A,M       ;IF THERE IS 1 THAN START HERE
        ORA     E         ;COMBINE -E- AND -A-
        MOV     E,A       ;STORE IN -E-
        POP     H         ;GET BACK PROGRAMS STATUS
        MOV     A,M       ;PUT STATUS IN -ACC-
        RRC               ;CHECK TO SEE IF THIS IS DIRECT ACCESS
        JNC     G38       ;IF IT IS GO HOME
```

```
        CMC
        RET
G38     RRC                 ;CHECK TO SEE IF THIS IS INDIRECT
                                                          ACCESS
        JC      G39         ;JUMP IF THERE IS ANY ACCESS DEFINITION
G47     LXI     H,M8        ;DISPLAY MODE? MESSAGE
        MVI     A,05H
        MOV     B,A
        CALL    MESSAG
        LXI     H,3FFFH
        CALL    BELL
        STC
        RET
G39     RRC                 ;CHECK TO SEE IF THIS BINARY DATA
        JNC     G40         ;JUMP IF IT ISNT
G52     LDA     1848H       ;CHECK TO SEE IF THIS IS DATA OR
                                                         ADDRESS
        ORA     A
        RP
        LXI     H,2000H ;ADD THE ADJUSTMENT TO THE ACTUAL DATA
        DAD     D
        ORA     A           ;RESET CY FLAG
        XCHG
        RET
G40     RRC                 ;CHECK TO SEE IF THIS IS DECIMAL DATA
        JNC     G40         ;JUMP IF THERE ISNT ANY DATA TYPE
                                                         DEFINED
        XCHG                ;WE NEED TO CONVERT THIS DECIMAL NUMBER
                                                     INTO BINARY
        CALL    DECBIN
        XCHG
        JMP     G52         ;JUMP TO ADJUST NEW BINARY NUMBER
INDRCT  LDA     1810H       ;THIS PROGRAM SETS OTHER PROGRAMS
                                              STATUS TO BE INDIRECT
        CPI     0BDH        ;IF THE LAST ALFA ENTERED WAS AN =
        JZ      G41         ;THEN JUMP
        LXI     D,180FH ;SET UP TO CHANGE ALL PROGRAM STATUS TO
                                                 INDIRECT ACCESS
        LXI     H,17FFH
G44     INX     H           ;POINT TO NEXT STATUS TO BE UPDATED
        MOV     A,M         ;MOVE STATUS INTO -ACC-
        ANI     0FEH        ;MASK OUT DIRECT ACCESS FLAG
        ORI     02H         ;MASK IN INDIRECT ACCESS FLAG
        MOV     M,A         ;STORE STATUS
        CALL    CMPDH       ;WAS THAT THE LAST STATUS UPDATE?
        JNZ     G44         ;JUMP IF IT WASNT
G45     CALL    ENDIT       ;UPDATING FINISHED GO HOME
        RET
G42     CALL    ERROR
G41     CALL    MANY        ;GET THE NEXT PROGRAM STATUS TO BE
                                                         UPDATED
        MOV     A,B         ;WAS THERE AN ENTRY IN SUB MANY?
```

```
        CPI     00H
        JZ      G41      ;JUMP IF THERE WASNT ANY ENTRY
        CALL    LOOKUP   ;SEE IF PROGRAM EXISTS
        JC      G42      ;JUMP IF IT DOESNT EXIST
        LXI     H,8000H  ;SEE IF PROGRAM NEEDS A STATUS
        CALL    CMPDH
        JNC     G42      ;JUMP IF NO STATUS IS NEEDED
        LDAX    D        ;MOVE STATUS INTO -ACC-
        ANI     0FEH     ;MASK OUT DIRECT FLAG
        ORI     02H      ;MASK IN INDIRECT FLAG
        STAX    D        ;STORE STATUS
        LDA     1810H    ;MOVE LSDIGIT INTO -ACC-
        CPI     0BDH     ;IS IT A =
        JZ      G41      ;GO GET ANOTHER PROGRAM TO UPDATE
        JMP     G45      ;JUMP TO END
BNRY    LDA     1810H    ;THIS PROGRAM CHANGES THE STATUS OF A
                                         PROGRAM TO ACCEPT
        CPI     0BDH     ;BINARY INFO
        JZ      G46      ;JUMP IF THE LSDIGIT WAS A =
        LXI     D,180FH  ;SET UP TO GET STATUS OF ALL PROGRAMS
                                        AND CHANGE THEM TO
        LXI     H,17FFH  ;BINARY
G48     INX     H        ;POINT TO NEXT PROGRAM STATUS TO BE
                                                   UPDATED
        MOV     A,M      ;MOVE STATUS INTO -ACC-
        ANI     0F7H     ;MASK OUT DECIMAL FLAG
        ORI     04H      ;MASK IN BINARY FLAG
        MOV     M,A      ;STORE STATUS
        CALL    CMPDH    ;WAS THAT THE LAST STATUS TO BE
                                                  UPDATED?
        JNZ     G48      ;JUMP IF IT WASNT
G49     CALL    ENDIT    ;FINISHED UPDATING GO HOME
        RET
G50     CALL    ERROR
G46     CALL    MANY     ;GET A PROGRAM
        MOV     A,B      ;WAS THERE AN ENTRY IN SUB MANY?
        CPI     00H
        JZ      G45      ;JUMP IF THERE WASNT ANY ENTRY
        CALL    LOOKUP   ;GET ITS STATUS LOCATION
        JC      G50      ;JUMP IF THERE WAS AN ERROR
        LXI     H,8000H  ;CHECK TO SEE IF PROGRAM NEEDS A STATUS
        CALL    CMPDH
        JNC     G50      ;JUMP IF NO STATUS IS NEEDED
        LDAX    D        ;MOVE STATUS INTO -ACC-
        ANI     0F7H     ;MASK OUT DECIMAL FLAG
        ORI     04H      ;MASK IN BINARY FLAG
        STAX    D        ;STORE STATUS
        LDA     1810H    ;MOVE LSDIGIT INTO -ACC-
        CPI     0BDH     ;SEE IF IT IS A =
        JZ      G46      ;JUMP IF IT IS AND GET ANOTHER PROGRAM
                                                 TO UPDATE
        JMP     G49      ;FINISHED UPDATING GO HOME
```

```
DCML    LDA     1810H   ;THIS SUB CHANGES PROGRAM STATUS TO BE
                                                        DECIMAL
        CPI     0BDH    ;WAS THE LSDIGIT A =?
        JZ      G51     ;JUMP IF IT WAS
        LXI     D,180FH ;SET UP TO CHANGE ALL PROGRAM STATUS TO
                                                     BE DECIMAL
        LXI     H,17FFH
G53     INX     H       ;POINT TO NEXT STATUS TO BE UPDATED
        MOV     A,M     ;MOVE STATUS INTO -ACC-
        ANI     0FAH    ;MASK OUT BINARY AND DIRECT FLAGS
        ORI     0AH     ;MASK IN DECIMAL AND INDIRECT FLAGS
        MOV     M,A     ;STORE STATUS
        CALL    CMPDH   ;IS THIS THE LAST STATUS UPDATE?
        JNZ     G53     ;JUMP IF IT ISNT
G54     CALL    ENDIT   ;UPDATING FINISHED GO HOME
        RET
G55     CALL    ERROR
G51     CALL    MANY    ;GET A PROGRAM
        MOV     A,B     ;WAS THERE AN ENTRY IN SUB MANY?
        CPI     00H
        JZ      G51     ;JUMP IF THERE WERE NO ENTRIES
        CALL    LOOKUP  ;GET THE PROGRAM STATUS
        JC      G55     ;JUMP IF THERE WAS AN ERROR
        LXI     H,8000H ;SEE IF THIS PROGRAM NEEDS A STATUS
        CALL    CMPDH
        JNC     G55     ;JUMP IF NO STATUS
        LDAX    D       ;MOVE STATUS INTO -ACC-
        ANI     0FAH    ;MASK OUT BINARY AND DIRECT FLAGS
        ORI     0AH     ;MASK IN DECIMAL AND INDIRECT FLAGS
        STAX    D       ;STORE STATUS
        LDA     1810H   ;MOVE LSDIGIT INTO THE -ACC-
        CPI     0BDH    ;SEE IF IT IS A =
        JZ      G51     ;JUMP AND GET ANOTHER PROGRAM
        JMP     G54     ;JUMP TO END
MOVEM   MVI     C,00H   ;THIS PROGRAM MOVES BLOCKS OF MEMORY
G65     INR     C       ;NUMBER-OF-ADDRESS COUNTER IS
                                                    INCREAMENTED
        MOV     A,C     ;FIND OUT WHAT ADDRESS THIS IS
        CPI     01H     ;IS IT THE FIRST ONE?
        JZ      G56     ;JUMP IF IT IS
        CPI     02H     ;IS IT THE SECOND ONE?
        JZ      G57     ;JUMP IF IT IS
        LXI     H,M3    ;IT MUST BE THE THIRD ADDRESS SO SET UP
                                              TO DISPLAY MESSAGE
G58     MVI     A,07H   ;PUT NUMBER OF CHARECTORS TO BE
                                              DISPLAYED IN -ACC-
        CALL    MESSAG
        LXI     H,1803H ;LOAD STATUS ADDRESS IN -HL- AND GET AN
                                                         ADDRESS
        CALL    ADDRES
        RC              ;RETURN IF THERE WAS AN ERROR IN THE
                                                             SUB
```

```
        PUSH    D           ;SAVE THIS ADDRESS
        MOV     A,C         ;SEE IF THIS IS THE LAST ADDRSS WE HAVE
                                                             TO GET
        CPI     03H
        JNZ     G65         ;JUMP IF IT ISNT THE LAST ONE
        POP     B           ;PUT THE NEW MIN ADDRESS IN -BC-
        POP     D           ;PUT THE OLD MAX ADDRESS IN -DE-
        POP     H           ;PUT THE OLD MIN ADDRESS IN -HL-
        CALL    CMPDH       ;SEE IF MAX IS LESS THAN MIN
        JNC     G63         ;JUMP IF IT ISNT
        LXI     H,M4        ;DISPLAY MIN>MAX MESSAGE
        MVI     A,07H
        CALL    MESSAG
        LXI     H,3FFFH ;ERROR SO RING BELL AND START OVER
                                                              AGAIN
        CALL    BELL
        LHLD    182AH       ;JUMP TO ESCAPE LOCATION
        PCHL
G63     CALL    MOVE        ;NOW MOVE THE DATA
        CALL    ENDIT       ;FINISHED SO DISPLAY END MESSAGE
        RET
G56     LXI     H,M1
        JMP     G58
G57     LXI     H,M2
        JMP     G58
        END
```

```
        STITLE  "BBIMS2 WRITTEN BY JIM MANLEY     THIRD OF SIX"
        GLOBAL  RECEV
        GLOBAL  TRNSMT
        GLOBAL  G48
        GLOBAL  LDBUFF
        GLOBAL  TTYLNK
        GLOBAL  MOVE
        GLOBAL  FEPROM
        GLOBAL  BIN
        GLOBAL  M4
        GLOBAL  M5
        GLOBAL  M6
        GLOBAL  M7
        GLOBAL  M9
        GLOBAL  M11
        GLOBAL  TRMOUT
        GLOBAL  ERROR
        GLOBAL  CLEAR
        GLOBAL  READ
        GLOBAL  DISPL
        GLOBAL  COMPA
        GLOBAL  COMPD
        GLOBAL  ALTR
        GLOBAL  GETDAT
        GLOBAL  ALREAD
        GLOBAL  ADDRES
        GLOBAL  BELL
        GLOBAL  MESSAG
        GLOBAL  DSADDR
        GLOBAL  DSDATA
        GLOBAL  NMREAD
        GLOBAL  CMPDH
        GLOBAL  ENDIT
        GLOBAL  DECBIN
COMPA   MVI     A,0FFH  ;IDENTIFY AS MEMORY VERSES MEMORY
        STA     184EH
        JMP     G123
COMPD   MVI     A,00H   ;IDENTIFY AS MEMORY VERSES DATA
        STA     184EH
G123    MVI     C,00H   ;THIS SUB COMPARES A BLOCK OF MEMORY TO
                                                  A BYTE OR TO
G115    INR     C       ;ANOTHER BLOCK OF MEMORY,-C- IS
                                            COUNTING HOW MANY
        MOV     A,C     ;ADDRESS WE HAVE,NOW WE ARE CHECK WHICH
                                        ADDRESS WE ARE GETTING
        CPI     01H     ;IS IT THE FIRST ONE?
        JZ      G116    ;JUMP IF IT IS,AND SET UP TO DISPLAY
                                                       MESSAGE
        LXI     H,M6    ;IT MUST BE THE SECOND ADDRESS SO SET
                                         UP TO DISPLAY MESSAGE
G117    MVI     A,08H
        CALL    MESSAG
```

```
        LXI     H,1805H ;LOAD -HL- WITH STATUS ADDRESS
        CALL    ADDRES
        JC      G145    ;RETURN IF THERE WAS AN ERROR IN SUB
        PUSH    D       ;SAVE ADDRESS
        MOV     A,C     ;CHECK IF THIS IS THE LAST ADDRESS WE
                                                    HAVE TO GET
        CPI     02H
        JNZ     G115    ;JUMP IF ITS NOT AND GET ANOTHER
                                                        ADDRESS
        LXI     H,M11   ;DISPLAY AGAINST MESSAGE
        MVI     A,07H
        CALL    MESSAG
        LDA     184EH   ;IS THIS MEMORY VERSES MEMORY OR VERSES
                                                          DATA?
        ORA     A
        JM      G120    ;JUMP IF IT IS MEMORY VERSES MEMORY
        LXI     H,1805H ;LOAD -HL- WITH STATUS ADDRESS
        CALL    GETDAT  ;GET A DATA BYTE
        JC      G145    ;RETURN IF THERE WAS INCORRECT MODE
        POP     D       ;GET BACK MAX ADDRESS AND PUT IN -DE-
        POP     H       ;GET BACK MIN ADDRESS AND PUT IN -HL-
        PUSH    PSW     ;SAVE -ACC-
        CALL    CMPDH   ;IS MIN GREATER THAN MAX?
        JC      G122    ;JUMP IF IT WAS AND DISPLAY MIN>MAX
                                                        MESSAGE
G131    POP     PSW     ;GET BACK DATA
        CMP     M       ;IS (HL)=-ACC- ?
        PUSH    H
        POP     B
        PUSH    PSW
        MOV     A,M
        CNZ     COMPSB  ;CALL IF IT ISNT AND DISPLAY ADDRESS
                                                       AND DATA
        CALL    CMPDH   ;IS THIS THE LAST COMPARISON?
        JNZ     G124    ;GO AND POINT TO NEXT COMPARISON
                                                       LOCATION
        POP     PSW
G126    CALL    ENDIT   ;FINISH UP
        RET
G124    INX     H
        JMP     G131
G122    POP     PSW     ;DISPLAY MIN>MAX MESSAGE
        LXI     H,M4
        MVI     A,07H
G118    CALL    MESSAG
        LXI     H,3FFFH ;RING BELL
        CALL    BELL
G145    LHLD    182AH
        PCHL
G136    CALL    CLEAR
        JMP     G145
G116    LXI     H,M5
```

```
        JMP     G117
G120    LXI     H,1805H ;LOAD -HL- WITH STATUS ADDRESS
        CALL    ADDRES  ;GET AN ADDRESS
        JC      G145    ;RETURN IF THERE WAS AN ERROR IN SUB
        MOV     B,D     ;MOVE -DE- INTO -BC-
        MOV     C,E
        POP     D       ;GET BACK MAX AND PUT IT IN -DE-
        POP     H       ;GET BACK MIN AND PUT IT IN -HL-
        CALL    CMPDH   ;CHECK IF MIN>MAX
        JC      G122    ;JUMP IF MIN>MAX
G130    LDAX    B       ;MOVE WHATS IN -BC- AND COMPARE IT TO
                                                 WHATS IN -HL-
        CMP     M
        CNZ     COMPSB  ;CALL IF THEY ARENT EQUAL AND DISPLAY
                                              DATA AND ADDRESS
        CALL    CMPDH   ;CHECK IF THIS IS THE LAST ONE
        JZ      G126    ;FINISH UP
        INX     B       ;POINT TO NEXT LOCATION TO BE CHECKED
        INX     H
        JMP     G130
COMPSB  PUSH    H       ;THIS SUB DISPLAYS DATA AND ADDRESS AND
                                                WAITS FOR A CR
        PUSH    D
        PUSH    PSW
        CALL    CLEAR   ;CLEAR ALL DISPLAYS FIRST
        MOV     D,B     ;MOVE -BC- INTO -DE- TO BE DISPLAYED
        MOV     E,C
        LXI     H,1805H ;LOAD -HL- WITH STATUS ADDRESS
        CALL    DSADDR  ;DISPLAY THE ADDRESS
        PUSH    PSW
        MVI     A,0A0H  ;OUTPUT A SPACE
        CALL    TRMOUT
        POP     PSW
        CALL    DSDATA  ;DISPLAY DATA
G134    PUSH    B       ;WAIT IN LOOP UNTIL SPACE
        CALL    READ
        POP     B
        CPI     9BH     ;IS IT AN ESC , IF IT IS THEN GO HOME
        JZ      G136
        CPI     8DH
        JNZ     G134
        POP     PSW
        POP     D
        POP     H
        RET
ALTR    MVI     A,07H   ;THIS SUB ALTERS A MEMORY LOCATION
        LXI     H,M5    ;SET UP TO DISPLAY ADDRESS MESSAGE
        LXI     B,0004H ;OFFSET THIS MESSAGE
        [illegible]
        [illegible] MESSAG
        [illegible]     ;LOAD -HL- WITH STATUS ADDRESS
```

```
         JC      G99       ;JUMP IF THERE WAS AN ERROR IN SUB
         LXI     H,1801H
G95      CALL    CLEAR     ;DISPLAY THE ADDRESS
         CALL    DSADDR
         MVI     A,0A0H    ;OUTPUT A SPACE
         CALL    TRMOUT
G104     LDAX    D         ;LOAD -ACC- WITH DATA
         CALL    DSDATA    ;DISPLAY THE DATA
         MVI     A,0ADH
         CALL    TRMOUT
G105     PUSH    D
         MVI     A,0FFH    ;SET UP FLAG TO TELL SUB GETDAT TO
                                                         RETURN
         STA     1849H
         CALL    GETDAT    ;GET DATA
         JC      G96       ;JUMP IF MODE ERROR
         POP     D
         STAX    D         ;STORE NEW DATA
G97      INX     D         ;POINT TO NEW LOCATION TO CHANGE
         JMP     G95
G96      LDA     1849H     ;CHECK TO SEE IF MODE ERROR OCCURED
         ORA     A
         JM      G100      ;JUMP IF ERROR
         POP     D
         JMP     G97       ;JUMP TO POINT TO NEXT ADDRESS
G99      MOV     A,B       ;CHECK IF MODE ERROR OCCURED IN SUB
         CPI     05H
         JZ      G152      ;JUMP IF IT IS
         LXI     H,M9      ;DISPLAY ERROR MESSAGE
         MVI     A,05H
G151     CALL    MESSAG
G152     LXI     H,3FFFH   ;SET UP TO RING BELL
         CALL    BELL
G150     LHLD    182AH
         PCHL
G100     XRA     A         ;CLEAR GETDAT FLAG
         STA     1849H
         JMP     G150
GETDAT   PUSH    H         ;THIS SUB GET A DATA BYTE,ALSO CAN
                                        RETURN UPON 0000 ENTRY
         XRA     A         ;SET UP TO CLEAR ADDRESS FLAG IN
                                                    ADDRESS SUB
         STA     1848H
         CALL    ADDRES    ;GET THE DATA
         MVI     A,0FFH    ;SET UP TO SET ADDRESS FLAG IN ADDRESS
                                                            SUB
         STA     1848H
         JC      G139      ;JUMP IF THERE WAS AN ERROR IN SUB
         MOV     A,D       ;CHECK TO SEE IF MSBYTE OF -DE- IS 00
         CPI     00H
         MOV     A,E
         JZ      G133      ;JUMP IF GOOD DATA
```

```
        LXI     H,M7 ;SET UP TO DISPLAY BAD DATA MESSAGE
        MVI     A,08H
        CALL    MESSAG
        LXI     H,3FFFH ;SET UP TO RING BELL
        CALL    BELL
        MVI     B,04H   ;SET -B- TO BE 4 FOR ERROR
G132    STC             ;SET THE CARRY TO SHOW GENERAL ERROR
G133    POP     H
        RET
G139    MOV     A,B     ;WAS THERE A MODE ERROR?
        CPI     05H
        JZ      G132    ;JUMP IF THERE WAS
        CPI     00H     ;SEE IF WE ARE TO RETURN ANY WAY
        JZ      G137    ;JUMP TO CHECK FLAG
G135    LXI     H,3FFFH ;SET UP TO RING BELL AND GO GET GOOD
                                                         DATA
        CALL    BELL
        POP     H
        JMP     GETDAT
G137    LDA     1849H   ;CHECK RETURN FLAG
        ORA     A
        JP      G135    ;GO RING BELL AND GET GOOD DATA BECAUSE
                                      NOT SUPPOSED TO RETURN
        XRA     A       ;CLEAR THIS FLAG FOR NEXT SUB
        STA     1849H
        JMP     G132    ;FINISH UP
DECBIN  PUSH    D       ;THIS SUB CONVERTS DECIMAL NUMBER IN
                                               -HL- INTO BINARY
        PUSH    B
        PUSH    PSW
        MOV     A,H     ;CHECK EACH DIGIT AND MAKE SURE IT HAS
                                               NO LETTERS IN IT
        ANI     0FH
        CPI     0AH
        JNC     G127
        MOV     A,H
        ANI     0F0H
        CPI     0A0H
        JNC     G127
        MOV     A,L
        ANI     0FH
        CPI     0AH
        JNC     G127
        MOV     A,L
        ANI     0F0H
        CPI     0A0H
        JNC     G127
        LXI     D,0000H ;SEE IF WORD IS 0000
        CALL    CMPDH
        JZ      G125    ;JUMP IF IT IS
        LXI     D,8191H ;SEE IF WORD IS GREATER THAN MAX MEMORY
        CALL    CMPDH
```

```
        JC      G127    ;JUMP IF IT IS AND DISPLAY ERROR
                                    MESSAGE THEN GO HOME
        PUSH    H
        LXI     H,0000H ;SET BINARY TO 0000
        SHLD    184AH   ;STORE BINARY EQUIVALENT
        LXI     H,BIN   ;POINT TO DECIMAL LOOK UP TABLE
        MOV     A,L     ;FIND END OF TABLE
        DCR     A
        STA     184FH   ;STORE FIRST LOCATION FOR LATER USE
        LXI     B,0019H
        DAD     B
        LXI     B,0000H ;SET NEW DECIMAL TO 0000
G101    MOV     D,M     ;GET DECIMAL FROM MEMORY
        DCX     H
        MOV     E,M
        DCX     H       ;NOW DECIMAL NUMBER IS IN -DE-
        XTHL            ;EXCHANGE NUMBER TO CONVERT AND POINTER
        PUSH    H       ;SAVE NUMBER TO CONVERT
        MOV     H,B
        MOV     L,C     ;PUT NEW DECIMAL NUMBER INTO -HL-
        MOV     A,L     ;ADD -HL- TO -DE- AND DECIMAL ADJUST
        ADD     E
        JC      G109
        DAA
        JNC     G110
G111    INR     H
G110    MOV     L,A
        MOV     A,H
        ADD     D
        DAA
        MOV     H,A
        XCHG            ;MOVE -HL- INTO -DE-
        POP     H       ;GET BACK NUMBER TO BE CONVERTED
        CALL    CMPDH   ;SEE IF THIS ADDITION DOESNT GO HIGHER
                              THAN NUMBER TO BE CONVERTED
        JZ      G102    ;HAVE TO SET THE CY BEFORE ROTATING
                                             INTO BINARY
        JNC     G138    ;JUMP IF IT WAS
G128    MOV     B,D
        MOV     C,E
G138    PUSH    H       ;SAVE NUMBER TO BE CONVERTED
        LHLD    184AH   ;GET BINARY EQUIVALENT THAT WE SAVED
        MOV     A,L     ;ROTATE IN THE CY BIT
        RAL
        MOV     L,A
        MOV     A,H
        RAL
        MOV     H,A
        SHLD    184AH   ;SAVE BINARY EQUIVALENT
        POP     H       ;GET BACK NUMBER TO BE CONVERTED BUT
                                  ONLY TO CORRECT STACK
        XTHL            ;NOW SWITCH NUMBER TO BE CONVERTED WITH
```

```
                                                          DECIMAL POINTER
         LDA     184FH   ;IS THIS THE LAST LOOKUP LOCATION?
         CMP     L
         JNZ     G101    ;JUMP IF THE COMPARISON A WHILE BACK
                                                                WAS EQUAL
         POP     H       ;CORRECT STACK
         LHLD    184AH   ;PUT BINARY CONVERSION INTO -HL-
G125     POP     PSW
         POP     B
         POP     D
         RET
G109     DAA
         INR     H
         JC      G111
         JMP     G110
G102     STC
         JMP     G128
G127     LXI     H,M9 ;DISPLAY ERROR AND GO HOME
         MVI     A,05H
         CALL    MESSAG
         LXI     H,3FFFH ;SET UP TO RING BELL
         CALL    BELL
         LHLD    182AH
         PCHL
DISPL    LDA     1855H   ;THIS PROGRAM DISPLAYS DATA LOCATED
                                                 BETWEEN TO ADDRESSES
         ORA     A
         JP      G171    ;IF CONSOLE IS ACTIVE DISPLAY ERROR
         MVI     C,00H
G83      INR     C       ;-C- IS NUMBER-OF-ADDRESS COUNTER
         MOV     A,C     ;FIND OUT WHICH ADDRESS WE ARE LOOKING
                                                                      FOR
         CPI     01H     ;IS IT THE FIRST ONE?
         JZ      G84     ;JUMP IF IT IS
         LXI     H,M6 ;SET UP TO DISPLAY MESSAGE
G85      MVI     A,08H
         CALL    MESSAG
G86      LXI     H,1800H ;PUT STATUS ADDRESS INTO -HL-
         CALL    ADDRES  ;GET AN ADDRESS
         JC      G94     ;JUMP IF THERE WAS AN ERROR IN SUB
G87      PUSH    D       ;SAVE ADDRESS
         MOV     A,C     ;IS THIS THE LAST ADDRESS WE HAVE TO
                                                                    GET ?
         CPI     02H
         JNZ     G83     ;JUMP IF IT ISNT
         POP     D       ;GET BACK MAX AND PUT IT IN -DE-
         POP     H       ;GET BACK MIN AND PUT IT IN -HL-
         CALL    CMPDH   ;MAKE SURE MIN ISNT GREATER THAN MAX
         JC      G90     ;JUMP IF MIN IS GREATER THAN MAX
         CALL    CLEAR
         XCHG            ;PUT MIN ADDRESS INTO -DE-
         MOV     A,L     ;SET LOW NIBBLE MAX TO F AND LOW NIBBLE
```

```
                                                            MIN TO 00
         ORI      0FH
         MOV      L,A
         MOV      A,E
         ANI      0F0H
         MOV      E,A
         PUSH     H
G174     LXI      H,1800H ;PUT STATUS ADDRESS INTO -HL-
         CALL     DSADDR
G173     MVI      A,0A0H  ;OUTPUT A SPACE
         CALL     TRMOUT
         IN       91H     ;IS ENTRY FROM TERMINAL PRESENT?
         RRC
         RRC
         JNC      G121    ;JUMP IF NO ENTRY
         IN       90H     ;CHECK FOR ESCAPE CHARECTOR
         ORI      80H
         CPI      9BH
         JZ       G93     ;IF ESCAPE THEN TERMINATE PROGRAM
G129     IN       91H     ;LOOK FOR NEXT ENTRY,STAY IN LOOP UNTIL
                                                 ANOTHER ENTRY
         RRC
         RRC
         JNC      G129
         IN       90H
G121     LDAX     D       ;GET DATA TO BE DISPLAYED
         LXI      H,1800H ;PUT STATUS ADDRESS INTO -HL-
         CALL     DSDATA  ;DISPLAY DATA
         POP      H
         CALL     CMPDH   ;CHECK TO SEE IF THIS IS THE LAST ONE
         JZ       G93     ;JUMP IF IT IS THE LAST ONE
         PUSH     H       ;SAVE MAX LOCATION
         INX      D       ;POINT TO NEXT LOCATION TO BE DISPLAYED
         MOV      A,E     ;IF THE LSNIBBLE IS A 0 OR 8 THEN
                                                  OUTPUT CR LF
         ANI      0FH
         JZ       G172
         CPI      08H
         JNZ      G173
G172     MVI      A,8DH   ;OUTPUT A CR LF
         CALL     TRMOUT
         MVI      A,8AH
         CALL     TRMOUT
         JMP      G174
G171     CALL     ERROR   ;CALL ERROR IF CONSOLE IS ACTIVE
         RET
G93      CALL     ENDIT   ;FINISHED
         LHLD     182AH
         PCHL
G84      LXI      H,M5
         JMP      G85
G90      LXI      H,M4    ;SET UP TO DISPLAY MIN>MAX MESSAGE
```

```
         MVI    A,07H
         CALL   MESSAG
         LXI    H,3FFFH ;SET UP TO RING BELL
         CALL   BELL
         RET
G94      MOV    A,B     ;CHECK TO SEE IF MODE ERROR OCCURED
         CPI    05H
         RZ             ;RETURN IF MODE ERROR
         LXI    H,3FFFH ;RING BELL THEN TRY FOR A GOOD ADDRESS
         CALL   BELL
         JMP    G86
FEPROM   XRA    A       ;THIS PROG OUTPUTS DATA TO FAKE EPROM
         OUT    0A3H
         LXI    H,2400H ;MOVE FROM 2400 TO 24FF TO FE00
         LXI    D,24FFH
         LXI    B,0FE00H
         CALL   MOVE
         LXI    H,2500H
         LXI    D,25FFH
         LXI    B,0FF00H;MOVE FROM 2500 TO 25FF TO FF00
         CALL   MOVE
         MVI    A,01H
         OUT    0A3H
         CALL   ENDIT
         RET
TTYLNK   MVI    A,0FDH  ;PROGRAM FOR COMMUNICATIONS BETWEEN
                                          BALLOON AND GROUND
         STA    1828H   ;LOAD LOOP COUNTER WITH -3
         MVI    A,0B0H  ;LOAD TIMER WITH STARTING COUNT USED
                                 FOR TIMING MAX LENGTH OF TRANS
         OUT    0D3H
         MVI    A,0FFH
         OUT    0D2H
         OUT    0D2H
         LDA    1829H   ;USART OR TM LINK?
         RAL
         JC     G103    ;JUMP IF USART
         LHLD   19AEH
         MOV    A,M
G106     CPI    05H
         JNZ    G48     ;JUMP IF NOT AN ENQUIRE CHARECTOR
G78      LDA    19B1H   ;GET COMMAND BYTE
         CPI    01H     ;IS IT A PAGE CONSTRUCTION COMMAND?
         JZ     G79     ;JUMP IF IT IS
         CPI    02H     ;IS IT A BOOK RUN COMMAND?
         JZ     G80     ;JUMP IF IT IS
         CPI    04H     ;IS IT A PAGE RUN PROGRAM?
         JZ     G80     ;JUMP IF IT IS
         CPI    08H     ;IS IT A DUMP COMMAND?
         JZ     G81     ;JUMP IF IT IS
         CPI    20H     ;IS IT A WAIT COMMAND?
         JZ     G81     ;JUMP IF IT IS
```

```
        CPI     40H        ;IS IT A CONTINUE COMMAND?
        JZ      G81        ;JUMP IF IT IS
        CPI     10H        ;IS IT A GO TO COMMAND?
        JZ      G88        ;JUMP IF IT IS
G70     MVI     A,02H      ;TRANSMIT AN STX CHARECTOR
        CALL    TRNSMT
        MVI     A,03H
        CALL    TRNSMT
        MVI     A,02H
        CALL    TRNSMT
        MVI     A,03H
        CALL    TRNSMT
        MVI     A,02H
        CALL    TRNSMT
        MVI     A,03H
        CALL    TRNSMT
        MVI     A,40H      ;CLEAR COMMAND
        STA     19B1H
        MVI     A,03H
        STA     19B2H
        JMP     G48
G88     LXI     D,19BEH
        JMP     G72
G82     CALL    TRNSMT
        JMP     G48
G79     LXI     D,19E2H    ;GET END ADDRESS
        JMP     G72
G80     LXI     D,19B6H    ;GET END ADDRESS
        JMP     G72
G81     LXI     D,19B2H    ;GET END ADDRESS
G72     LXI     H,19B0H    ;GET STARTING ADDRESS
G71     MOV     A,M        ;GET A CHARECTOR
        CALL    TRNSMT     ;TRANSMIT CHARECTOR
        CALL    CMPDH      ;WAS THIS THE LAST TRANSMITION?
        INX     H
        JNZ     G71        ;JUMP IF IT WASNT THE LAST ONE
        LDA     1828H      ;GET LOOP COUNTER AND INCREAMENT
        INR     A
        STA     1828H
        JNZ     G78
        MVI     B,0B0H
G73     CALL    RECEV      ;GET BACK CHARECTORS OR ABORT OR REPEAT
        JC      G48        ;JUMP IF ERROR
        CPI     1BH        ;JUMP IF ABORT COMMAND
        JZ      G48
        CPI     15H        ;IS IT A REPEAT CHARECTOR
        JZ      G74        ;JUMP IF IT IS
        CPI     03H        ;IS IT AN ETX CHARECTOR?
        JZ      G76        ;JUMP IF IT IS
        CPI     30H        ;IS THERE DATA TO BE DISPLAYED?
        JC      G75        ;JUMP IF THERE ISNT
        CPI     40H
```

```
         JZ      G75
         OUT     90H
G75      INR     B
         JNZ     G73
G76      MVI     A,8DH
         CALL    TRMOUT
         MVI     A,8AH
         CALL    TRMOUT
         MVI     A,0AAH
         CALL    TRMOUT
         LDA     19B1H    ;IF WAIT COMMAND DONT RESET MESSAGE
         CPI     20H
         JZ      G48
         MVI     A,40H    ;CLEAR COMMAND DEFINITION
         STA     19B1H
         MVI     A,03H
         STA     19B2H
         JMP     G48
G74      MVI     A,0FDH   ;LOAD LOOP COUNTER WITH -3
         STA     1828H
         JMP     G78
G103     IN      0C1H
         RAR
         RAR
         JNC     G103
         IN      0C0H     ;GET CHARECTOR
         JMP     G106
         END
```

```
STITLE  "BB1MS3 WRITTEN BY JIM MANLEY     FORTH OF SIX"
GLOBAL  M45
GLOBAL  M43
GLOBAL  M44
GLOBAL  GOTOBB
GLOBAL  MAIN
GLOBAL  LOP
GLOBAL  RATIO
GLOBAL  M30
GLOBAL  M31
GLOBAL  M32
GLOBAL  AMU
GLOBAL  TIME
GLOBAL  MASK
GLOBAL  IDNUM
GLOBAL  RPAGE
GLOBAL  RBOOK
GLOBAL  NPAGE
GLOBAL  DUMP
GLOBAL  CONT
GLOBAL  WAIT
GLOBAL  M12A
GLOBAL  RECEV
GLOBAL  TRNSMT
GLOBAL  LDBUFF
GLOBAL  M4
GLOBAL  M5
GLOBAL  M6
GLOBAL  M7
GLOBAL  M13
GLOBAL  BINCON
GLOBAL  GETDAT
GLOBAL  FILL
GLOBAL  BELL
GLOBAL  FILLM
GLOBAL  TRMOUT
GLOBAL  SWITCH
GLOBAL  ERROR
GLOBAL  DECBIN
GLOBAL  BINDEC
GLOBAL  GOTO
GLOBAL  ADDRES
GLOBAL  CMPDH
GLOBAL  ENDIT
GLOBAL  MESSAG
GLOBAL  CLEAR
GLOBAL  DSDATA
GLOBAL  DSADDR
GLOBAL  READ
GLOBAL  BIN
GLOBAL  TERMIN
GLOBAL  G5B2
```

```
        GLOBAL  TRMOUT
        GLOBAL  BAUD
BINDEC  PUSH    H       ;THIS SUB CONVERTS BINARY WORDS INTO 4
                                                PLACE DECIMAL
        PUSH    B
        PUSH    PSW
        MVI     A,0F8H  ;THIS IS OUR LOOP COUNTER 8 FOR 8 BITS
                                                     PER BYTE
        STA     184FH   ;STORE IN BUFFER
        XRA     A       ;THIS IS TO DETERMINE IF MSBYTE HAS
                                              BEEN PROCCESSED
        STA     182CH
        LXI     H,BIN   ;LOAD -HL- WITH BEGINNING OF LOOK UP
                                                        TABLE
        LXI     B,0000H ;LOAD -BC- WITH DECIMAL ZERO
G188    MOV     A,E     ;-DE- CONTAINS BINARY DATA TO BE
                                                    CONVERTED
        RAR             ;CHECK FOR A SET BIT
        MOV     E,A
        JNC     G192    ;JUMP IF NOT SET
        MOV     A,C     ;GET DECIMAL EQUIVALENT
        ADD     M       ;ADD TO WHATS IN MEMORY AT LOCATION
                                                         -HL-
        JC      G189    ;JUMP IF WE NEED TO ADD TO NEXT BYTE
        DAA             ;ADJUST DECIMAL
        JNC     G190    ;JUMP IF NO CARRY OUT OF FIRST BYTE
G191    INR     B       ;ADD ONE TO -B- IF CARRY
G190    MOV     C,A     ;ADJUST DECIMAL
        INX     H       ;POINT TO MSBYTE OF EQUIVALENT
        MOV     A,B     ;GET MSBYTE OF DECIMAL EQUIVALENT
        ADD     M       ;ADD TO BYTE POINTED TO BY -HL-
        DAA             ;ADJUST -ACC-
        MOV     B,A
G193    INX     H       ;POINT TO NEXT BIT EQUIVALENT
        LDA     184FH   ;INCREAMENT BIT COUNTER
        INR     A
        STA     184FH
        JNZ     G188    ;JUMP AND CONTINUE WITH NEXT IF NOT
                                                         DONE
        MOV     E,D     ;MOVE MSBYTE OF BINARY INTO -E-
        LDA     182CH   ;JUMP IF THIS SECTION WAS ALEADY DONE
        RLC
        JNC     G198    ;JUMP IF IT WASNT
        MOV     D,B     ;MOVE DECIMAL NUMBER INTO -DE-
        MOV     E,C
        POP     PSW
        POP     B
        POP     H
        RET
G189    DAA             ;ADJUST DECIMAL
        INR     B       ;ADD ONE TO MSBYTE OF DECIMAL
        JC      G191    ;IF A CARRY OUT OF DECIMAL ADJUST
```

```
                                                   INCREAMENT -B-
        JMP     G190
G192    INX     H          ;CORRECT POINTER
        JMP     G193
G198    MVI     A,0FFH     ;SET FLAG THAT WE HAVE DONE PART ONE
        STA     182CH
        MVI     A,0F8H
        STA     184FH
        JMP     G188
GOTO    MVI     A,07H      ;THIS PROGRAM GIVES PROCCESSOR CONTROL
                                                AT LOCATION XXXX
        LXI     H,M5
        LXI     D,0004H    ;OFFSET THIS MESSAGE
        DAD     D
        CALL    MESSAG
        LXI     H,1807H
        CALL    ADDRES
        RC
        XCHG
        PCHL
DSDATA  PUSH    D          ;THIS SUB SEPARATES DATA INTO TO
                                        NIBBLES AND DISPLAYS
        PUSH    PSW        ;THEM ,IT ALSO CONVERTS IF NEEDED
        MOV     E,A        ;SAVE DATA IN -E-
        MOV     A,M        ;CHECK TO SEE IF CONVERTION IS NEEDED
        RRC
        RRC
        RRC                ;MOVE BINARY FLAG INTO -ACC-
        JC      G81        ;JUMP IF IT IS ALREADY IN BINARY
        MVI     D,00H      ;SET UP TO CONVERT -DE-
        CALL    BINDEC
        MOV     A,D        ;NOW SEPARATE INTO 3 NIBBLES AND
                                                     DISPLAY
        ANI     0FH        ;MASK OUT HIGH NIBBLE
        CALL    TRMOUT
        STA     3805H      ;DISPLAY IT
        MOV     A,E
        ANI     0F0H       ;MASK OUT LOW NIBBLE
        RRC
        RRC                ;MOVE HIGH NIBBLE INTO LOW NIBBLE
        RRC
        RRC
        CALL    TRMOUT
        STA     3806H      ;DISPLAY IT
        MOV     A,E
        ANI     0FH
        CALL    TRMOUT
        STA     3807H      ;DISPLAY IT
G213    POP     PSW
        POP     D
        RET
G81     MOV     A,E        ;ALREADY BINARY SO DISPLAY ONLY 2
```

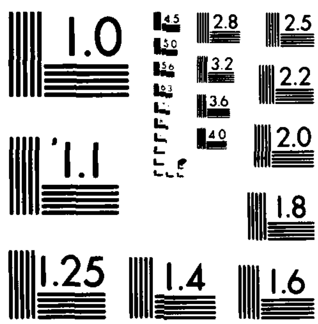
```
                                                 NIBBLES
        ANI     0F0H     ;MASK OUT HIGH NIBBLE
        RRC
        RRC
        RRC
        RRC
        CALL    TRMOUT
        STA     3806H    ;DISPLAY IT
        MOV     A,E
        ANI     0FH      ;MASK OUT LOW NIBBLE
        CALL    TRMOUT
        STA     3807H
        JMP     G213
DSADDR  PUSH    H        ;THIS SUB SEPARATES ADDRESS INTO 4
                                   NIBBLES AND DISPLAYS
        PUSH    D        ;THEM IT ALSO CONVERTS IF NEEDED
        PUSH    PSW
        MOV     A,M      ;PUT THE STATUS INTO THE -ACC-
        RRC              ;CHECK FOR CIRECT ACCESS FLAG
        JC      G82      ;JUMP IF NO ADJUSTMENT IS NEEDED
        LXI     H,0E000H;SUBTRACT 2000H FROM ADDRESS
        DAD     D
        XCHG
        RRC              ;IS DATA BINARY
        RRC
        JC      G82      ;JUMP IF IT IS
        CALL    BINDEC   ;CONVERT IF NEEDED
G82     MOV     A,D      ;WE WILL NOW DISPLAY -DE-
        ANI     0F0H     ;MASK OUT HIGH NIBBLE
        RRC
        RRC
        RRC
        RRC
        CALL    TRMOUT
        STA     3800H    ;DISPLAY IT
        MOV     A,D
        ANI     0FH      ;MASK OUT LOW NIBBLE
        CALL    TRMOUT
        STA     3801H    ;DISPLAY IT
        MOV     A,E
        ANI     0F0H     ;MASK OUT HIGH NIBBLE
        RRC
        RRC
        RRC
        RRC
        CALL    TRMOUT
        STA     3802H    ;DISPLAY IT
        MOV     A,E
        ANI     0FH      ;MASK OUT LOW NIBBLE
        CALL    TRMOUT
        STA     3803H    ;DISPLAY IT
        POP     PSW
```

```
        POP     D
        POP     H
        RET
BAUD    LXI     H,M13   ;THIS SUB CHANGES THE BAUD RATE FOR
                                                TERMINAL USE
        MVI     A,04H   ;SET UP TO DISPLAY RATE
        CALL    MESSAG
        LXI     H,1850H ;LOAD -HL- WITH FAKE STATUS ADDRESS
        MVI     A,05H   ;SET DIRECT AND BINARY FLAGS
        MOV     M,A     ;STORE STATUS
        CALL    ADDRES  ;GET BAUD RATE
        JC      G28     ;JUMP IF ERROR
        LXI     B,1280  ;LOAD DIVIDER FOR 75 BAUD
        LXI     H,0075H
        CALL    CMPDH
        JZ      G171    ;JUMP IF 75 BAUD
        LXI     B,873   ;110 BAUD
        LXI     H,0110H
        CALL    CMPDH
        JZ      G171
        LXI     B,320   ;300 BAUD
        LXI     H,0300H
        CALL    CMPDH
        JZ      G171
        LXI     B,160   ;600 BAUD
        LXI     H,0600H
        CALL    CMPDH
        JZ      G171
        LXI     B,80    ;1200 BAUD
        LXI     H,1200H
        CALL    CMPDH
        JZ      G171
        LXI     B,40    ;2400 BAUD
        LXI     H,2400H
        CALL    CMPDH
        JZ      G171
        LXI     B,20    ;4800 BAUD
        LXI     H,4800H
        CALL    CMPDH
        JZ      G171
        LXI     B,10    ;9600 BAUD
        LXI     H,9600H
        CALL    CMPDH
        JNZ     G28
G171    MVI     A,3EH   ;SET UP TO TURN ON SYSTEM USART CLOCK
        OUT     0D3H
        MOV     A,C     ;OUTPUT DIVIDER
        OUT     0D0H
        MOV     A,B
        OUT     0D0H
        CALL    ENDIT
        RET
```

AD-A115 399 NORTHEASTERN UNIV BOSTON MASS ELECTRONICS RESEARCH LAB F/G 7/4
CONTROL ELECTRONICS FOR AIR-BORNE QUADRUPOLE ION MASS SPECTROME--ETC(U)
OCT 81 J S ROCHEFORT, R SUKYS F19628-78-C-0218
UNCLASSIFIED AFGL-TR-82-0056 NL

4 of 4

END
DATE
FILMED
7-82
DTIC



MICROCOPY RESOLUTION TEST CHART

```
G28     CALL    ERROR
        RET
SWITCH  LDA     1855H   ;THIS SUB SWITCHES THE OUTPUT AND INPUT
                         DEVICE TO THE OPPOSITE OF WHICH IT WAS
        RAL
        CMC
        RAR
        STA     1855H
        IN      0B2H    ;NOW COMPLIAMENT CONSOLE LED
        ANI     02H     ;REMOVE ALL BITS BUT THE 2ND
        CMA             ;COMPLIAMENT BIT 2
        ANI     02H
        RRC             ;ROTATE INTO LSBIT
        ORI     02H
        OUT     0B3H
        CALL    ENDIT   ;FINISHED GO HOME
        RET
ASCONV  CPI     0B0H    ;THIS SUB CONVERTS ASCII TO SYSTEM
                                                          BINARY
        RC              ;IS IT GREATER THAN OR EQUAL TO A 0?
        CPI     0BAH    ;IS IT LESS THAN AN A?
        JNC     G8      ;JUMP IF ITS NOT
        SUI     0B0H    ;CONVERT TO BINARY 0 TO 9
        RET
G8      CPI     0C1H    ;IS IT GREATER THAN OR EQUAL TO AN A?
        RC              ;RETURN IF IT ISNT
        CPI     0C7H    ;IS IN LESS THAN A G?
        RNC             ;RETURN IF IT ISNT
        SUI     0B7H    ;CONVERT TO BINARY A TO F
        RET
BINCON  RAL             ;THIS SUB CONVERTS SYSTEM BINARY TO
                                                           ASCII
        JC      G29     ;JUMP IF MSBIT IS SET THIS MEANS
                                               ALREADY IN ASCII
        RAR
        CPI     0AH     ;IS IT LESS THAN A?
        JNC     G31     ;JUMP IF ITS NOT
        ADI     0B0H    ;CONVERT TO ASCII 0 TO 9
        RET
G29     RAR
        RET
G31     ADI     0B7H    ;CONVERT TO ASCII A TO F
        RET
FILLM   MVI     C,00H   ;THIS PROGRAM FILLS DEFINED MEMORY WITH
                                                    DEFINED DATA
G66     INR     C       ;THIS TELLS YOU WHAT ADDRESS POSITION
                                                 YOU ARE FILLING
        MOV     A,C     ;FIND OUT WHICH ONE YOUR FILLING AND
                                                 DISPLAY MESSAGE
        CPI     01H     ;IS IT THE FIRST ONE ?
        JZ      G67     ;JUMP IF IT IS
        LXI     H,M6
```

```
G69     MVI     A,08H
        CALL    MESSAG
G71     LXI     H,1802H ;PUT STATUS ADDRESS IN -HL-
        CALL    ADDRES  ;GET AN ADDRESS
        JC      G72     ;JUMP IF THERE WAS AN ERROR IN THE SUB
        PUSH    D       ;SAVE THE ADDRESS
        MOV     A,C     ;FIND OUT IF THIS IS THE LAST ADDRESS
                                                          NEEDED
        CPI     02H
        JNZ     G66     ;JUMP IF IT ISNT THE LAST ONE
        LXI     H,M7    ;SET UP TO DISPLAY DATA MESSAGE
        LXI     D,0004H ;OFFSET THIS MESSAGE
        DAD     D
        MVI     A,04H
        CALL    MESSAG
        LXI     H,1802H ;SET UP TO GET A DATA BYTE
        CALL    GETDAT
        JC      G79
        POP     D       ;PUT MAX ADDRESS INTO -DE-
        POP     H       ;PUT MIN ADDRESS INTO -HL-
        PUSH    PSW     ;SAVE FILLER
        CALL    CMPDH   ;SEE IF MAX IS LESS THAN MIN
        JC      G77     ;JUMP IF IT IS
        POP     PSW     ;GET BACK FILLER
G74     CALL    FILL    ;FILL MEMORY WITH FILLER
        CALL    ENDIT
        RET
G79     LHLD    182AH   ;SET UP TO ESCAPE
        PCHL
G77     LXI     H,M4    ;SET UP TO DISPLAY MIN>MAX
        MVI     A,07H
        CALL    MESSAG
        LXI     H,3FFFH
        POP     PSW
        CALL    BELL
        RET
G67     LXI     H,M5    ;JUMP TO DISPLAY MESSAGE
        JMP     G69
G72     MOV     A,B     ;FIND OUT WHAT ERROR IS PENDING
        CPI     05H     ;IS IT A MODE ERROR?
        JZ      G79     ;RETURN IF IT IS
        CPI     00H     ;IS IT A 0000 ADDRESS?
        JZ      G73     ;JUMP TO FIND OUT IF STANDARD FILL IS
                                                     TO BE USED
G68     LXI     H,3FFFH ;SET UP TO RING BELL AND TRY FOR A GOOD
                                                         ADDRESS
        CALL    BELL
        JMP     G71
G73     MOV     A,C     ;IF ADDRESS COUNTER IS 1 THEN STANDARD
                                                   FILL IS USED
        CPI     01H
        JNZ     G68     ;TRY FOR A GOOD ADDRESS
```

```
        LXI     H,2000H ;SET UP FOR STANDARD FILL
        LXI     D,27FFH
        MVI     A,0FFH
        JMP     G74     ;JUMP TO FILL
TERMIN  IN      91H
        RAR
        RAR
        JNC     TERMIN  ;JUMP IF NO DATA
        IN      90H     ;READ IN DATA
        ORI     80H
        PUSH    B
        MOV     B,A     ;SAVE DATA
        LDA     18EAH   ;GET RETURN FLAG
        RAL
        MOV     A,B
        POP     B
        CNC     ASCONV  ;CALL IF NOT SET
        JMP     G5B2
TRMOUT  PUSH    B       ;THIS SUB DISPLAYS -ACC- ON TERMINAL
        PUSH    PSW
        LDA     1855H   ;IS TERMINAL ACTIVE?
        ORA     A
        JP      G17     ;JUMP IF IT ISNT ACTIVE
        CALL    BINCON  ;CONVERT TO ASCII
        MOV     B,A
G181    IN      91H     ;TEST FOR ERRORS AND READINESS
        RAR
        JNC     G181    ;JUMP IF NOT READY
        MOV     A,B
        OUT     90H     ;DISPLAY
G17     POP     PSW
        POP     B
        RET
TRNSMT  PUSH    B       ;THIS SUB TRANSMITS DATA TO BALLOON
        MOV     B,A     ;SAVE DATA
G1      IN      0C1H    ;WAIT FOR TRANSMITTER READY
        RAR
        JNC     G1      ;JUMP IF TRANSMITTER NOT READY
        MOV     A,B
        OUT     0C0H    ;TRANSMIT DATA
        POP     B
        RET
RECEV   PUSH    H       ;THIS SUB RECEIVES DATA FROM BALLOON
        PUSH    D
G2      IN      91H     ;CHECK IF THERE IS TIME LEFT
        RLC
        JNC     G3
        LDA     1829H   ;WHICH TRANSMITTION MEDIA IS USED?
        RAL
        JC      G7      ;JUMP IF MEDIA IS USART
        CALL    LDBUFF  ;TRANSITION MEDIA IS TM
        MOV     A,M     ;GET NEW BUFFER DATA AND CHECK FOR SYNC
```

```
        CPI     90H
        JZ      G83     ;JUMP IF IN SYNC
        LHLD    182AH
        LXI     SP,1C00H
        PUSH    H
        PCHL
G83     LDA     196EH   ;GET STATUS BYTE
        ANI     04H     ;IS THERE DATA IN TTY LOCATION?
        JZ      G2      ;JUMP IF NO DATA YET
        LHLD    19AEH   ;GET LOCATION OF TTY BYTE
        MOV     A,M     ;GET TTY DATA
G6      POP     D
        POP     H
        ORA     A
        RET
G3      STC             ;SET ERROR FLAG
        POP     D
        POP     H
        RET
G7      IN      0C1H    ;CHECK DSR
        RAL
        JNC     G3
        RAR
        RAR
        RAR             ;CHECK RECIEVER READY
        JNC     G2      ;JUMP IF NOT READY
        IN      0C0H    ;GET DATA
        JMP     G6
CNVRT   MOV     A,H     ;THIS SUB CONVERTS -HL- INTO FOUR ASCII
                                                    CHARECTORS
        ANI     0F0H    ;DO TOP NIBBLE FIRST
        RRC
        RRC
        RRC
        RRC
        CALL    BINCON  ;CONVERT TO ASCII
        STAX    D       ;STORE IN -DE-
        INX     D
        MOV     A,H     ;DO NEXT NIBBLE
        ANI     0FH
        CALL    BINCON
        STAX    D
        INX     D
CNVRT1  MOV     A,L     ;IF ENTERED HERE WILL CONVERT -L- INTO
                                                    TWO CHAR
        ANI     0F0H
        RRC
        RRC
        RRC
        RRC
        CALL    BINCON
        STAX    D
```

```
        INX     D
        MOV     A,L
        ANI     0FH
        CALL    BINCON
        STAX    D
        INX     D
        RET
RPAGE   MVI     A,0FFH  ;THIS SUB SETS UP TO RUN A PAGE DURING
                                                        MAIN
        JMP     G5
RBOOK   XRA     A       ;THIS SUB SETS UP TO RUN A BOOK DURING
                                                        MAIN
G5      STA     182FH   ;THIS BUFFER TELLS IF BOOK OR PAGE IS
                                                   TO BE RUN
        LXI     H,M12A  ;SET UP TO DISPLAY ADDRESS MESSAGE
        MVI     A,07H
        CALL    MESSAG
        LXI     H,184FH ;LOAD -HL- WITH FAKE STATUS LOCATION
        MVI     A,05H   ;SET STATUS FOR DIRECT BINARY
        MOV     M,A
        CALL    ADDRES  ;GET LOCATION OF PAGE OR BOOK
        RC              ;RETURN IF ERROR
        LXI     H,19B0H ;LOAD -HL- WITH BEGINNING OF BUFFER TO
                                             STORE THIS DATA
        MVI     A,02H   ;STORE A STX CHARECTOR
        MOV     M,A
        INX     H
        LDA     182FH   ;IS THIS RUNNING A BOOK OR PAGE?
        RLC
        MVI     A,04H   ;LOAD -ACC- WITH PAGE DEFINITION
        JC      G9      ;JUMP IF IT IS A BOOK
        MVI     A,02H   ;LOAD -ACC- WITH PAGE DEFINITION
G9      MOV     M,A     ;STORE COMMAND DEFINITION
        INX     H
        XCHG
        CALL    CNVRT   ;NOW CONVERT AND STORE ADDRESS
        XCHG
        MVI     A,03H   ;STORE ETX CHARECTOR
        MOV     M,A
        JMP     MAIN
NPAGE   MVI     A,02H   ;THIS SUB IN CONJUNCTION WITH OTHER
                                                   FUNCTIONS
        STA     19B0H   ;CONSTRUCT A PAGE AND SET IT UP TO BE
                                                 RUN IN MAIN
        MVI     A,01H   ;STORE STX THEN COMMAND DEFINITION
        STA     19B1H
        LXI     D,19B2H
        LHLD    2400H
        CALL    CNVRT
        LHLD    2402H
        CALL    CNVRT
        LDA     2404H
```

```
        MOV     L,A
        CALL    CNVRT1
        LHLD    2405H
        CALL    CNVRT
        LHLD    2407H
        CALL    CNVRT
        LXI     B,2409H
G84     LDAX    B
        MOV     L,A
        CALL    CNVRT1
        MOV     A,C
        INX     B
        CPI     15H
        JNZ     G84
        LHLD    2416H
        CALL    CNVRT
        MVI     A,03H    ;NOW STORE ETX
        STA     19E2H
        JMP     MAIN
DUMP    MVI     A,08H    ;STORE COMMAND FOR DUMP DURING MAIN
G10     STA     19B1H
        MVI     A,02H    ;STORE STX
        STA     19B0H
        MVI     A,03H    ;STORE ETX
        STA     19B2H
        JMP     MAIN
CONT    MVI     A,40H    ;STORE COMMAND FOR CONTINUE DURING MAIN
        JMP     G10
WAIT    MVI     A,20H    ;STORE COMMAND FOR WAIT DURING MAIN
        JMP     G10
RATIO   LXI     H,M31    ;THIS SUB DEFINES RATIO FOR USE WITH
                                                     NPAGE
        MVI     A,06H    ;DISPLAY ADDRESS MESSAGE
        CALL    MESSAG
        LXI     H,1809H  ;LOAD -HL- WITH STATUS ADDRESS
        CALL    ADDRES   ;GET RATIO
        JC      RATIO    ;JUMP IF ERROR IN SUB
        LDA     1809H
        ANI     02H
        JZ      G4
        LXI     H,0E000H
        DAD     D
        XCHG
G4      MOV     A,E
        STA     2407H
        MOV     A,D
        STA     2408H
        CALL    ENDIT
        RET
MASK    LXI     H,M31    ;THIS SUB DEFINES MASK FOR USE WITH
                                                     NPAGE
        MVI     A,06H    ;DISPLAY NUMBER MESSAGE
```

```
         CALL    MESSAG
         LXI     H,1809H ;LOAD -HL- WITH STATUS ADDRESS
         CALL    GETDAT  ;GET MASK
         JC      MASK    ;JUMP IF ERROR IN SUB
         STA     2414H
         CALL    ENDIT
         RET
IDNUM    LXI     H,M31   ;THIS SUB DEFINES PROGRAM ID FOR USE
                                                    WITH NPAGE
         MVI     A,06H   ;DISPLAY NUMBER MESSAGE
         CALL    MESSAG
         LXI     H,1809H ;LOAD -HL- WITH STATUS ADDRESS
         CALL    ADDRES  ;GET IDNUM
         JC      IDNUM   ;JUMP IF ERROR IN SUB
         MOV     A,E
         STA     2416H
         MOV     A,D
         STA     2417H
         CALL    ENDIT
         RET
LOP      LXI     H,M31   ;THIS SUB DEFINES NUMBER OF LOOPS USED
                                                    WITH NPAGE
         MVI     A,06H
         CALL    MESSAG  ;DISPLAY NUMBER MESSAGE
         LXI     H,1809H ;LOAD -HL- WITH STATUS ADDRESS
         CALL    GETDAT  ;GET NUMBER OF LOOPS
         RC              ;RETURN IF ERROR
         STA     2404H
         CALL    ENDIT
         RET
TIME     LXI     H,M31   ;THIS SUB DEFINES TIME INTERVAL USED
                                                    WITH NPAGE
         MVI     A,06H
         CALL    MESSAG  ;DISPLAY NUMBER MESSAGE
         LXI     H,1809H ;LOAD -HL- WITH STATUS ADDRESS
         CALL    ADDRES  ;GET TIME
         RC              ;RETURN IF ERROR
         LDA     1809H
         ANI     02H
         JZ      G12
         LXI     H,0E000H
         DAD     D
         XCHG
G12      MOV     A,E
         STA     2405H
         MOV     A,D
         STA     2406H
         CALL    ENDIT
         RET
AMU      LXI     H,M30   ;THIS SUB DEFINES START AND STOP AMU
                                                USED WITH NPAGE
         MVI     A,06H   ;DISPLAY START? MESSAGE
```

```
        CALL    MESSAG
        CALL    READ     ;GET A YES OR NO ANSWER
        CPI     9BH      ;IS IT AN ESC?
        RZ               ;RETURN IF IT IS
        CPI     0CEH     ;IS IT AN N?
        JZ      G11      ;JUMP AND BYPASS DEFINING START AMU
        CPI     0D9H     ;IS IT A YES?
        JNZ     AMU      ;JUMP IF ITS NOT
        LXI     H,M31    ;DISPLAY NUMBER MESSAGE
        MVI     A,06H
        CALL    MESSAG
        LXI     H,1809H  ;LOAD -HL- WITH STATUS ADDRESS
        CALL    ADDRES   ;GET START AMU
        JC      AMU      ;JUMP IF ERROR IN SUB
        LDA     1809H
        ANI     02H
        JZ      G13
        LXI     H,0E000H
        DAD     D
        XCHG
G13     BYTE    18H,18H  ;ROTATE -DE- TO THE LEFT 2 TIMES
        LXI     H,1000H
        CALL    CMPDH
        JNC     AMU
        MOV     A,E
        ANI     0FCH     ;REMOVE 2 LSBITS
        STA     2400H
        MOV     A,D
        STA     2401H
G11     LXI     H,M32    ;DISPLAY ENDING?
        MVI     A,07H
        CALL    MESSAG
        CALL    READ     ;GET A YES OR NO ANSWER
        CPI     9BH      ;IS IT AN ESC?
        RZ               ;RETURN IF IT IS
        CPI     0CEH     ;IS IT A NO?
        JZ      G80      ;RETURN IF IT IS
        CPI     0D9H     ;IS IT A YES?
        JNZ     G11      ;JUMP IF ITS NOT
        LXI     H,M31    ;DISPLAY NUMBER MESSAG
        MVI     A,06H
        CALL    MESSAG
        LXI     H,1809H  ;LOAD -HL- WITH STATUS ADDRESS
        CALL    ADDRES   ;GET STOP AMU
        JC      G11      ;RETURN IF ERROR
        LDA     1809H
        ANI     02H
        JZ      G14
        LXI     H,0E000H
        DAD     D
        XCHG
G14     BYTE    18H,18H  ;ROTATE -DE- 2 TIMES TO THE LEFT
```

```
        LXI     H,1000H
        CALL    CMPDH
        JNC     G11
        MOV     A,E
        ORI     03H      ;SET 2 LSBITS
        STA     2402H
        MOV     A,D
        STA     2403H
G80     CALL    ENDIT
        RET
GOTOBB  LXI     H,M45    ;JUMPS TO REP,PROG,INST IN BBIMS
        MVI     A,08H    ;DISPLAY RPETOIRE MESSAGE
        CALL    MESSAG
        LXI     H,1809H  ;GET ADDRESS OF REP
        CALL    ADDRES
        JC      GOTOBB   ;JUMP BACK IF ERROR
        LXI     H,19B2H  ;CONVERT AND STORE
        XCHG
        CALL    CNVRT
        XCHG
G15     LXI     H,M43    ;DISPLAY PROGRAM MESSAGE
        MVI     A,07H
        CALL    MESSAG
        LXI     H,1809H
        CALL    ADDRES   ;GET ADDRESS OF PROGRAM
        JC      G15      ;JUMP BACK IF ERROR
        LXI     H,19B6H  ;CONVERT AND STORE
        XCHG
        CALL    CNVRT
        XCHG
G16     LXI     H,M44    ;DISPLAY INSTRCT MESSAGE
        MVI     A,07H
        CALL    MESSAG
        LXI     H,1809H
        CALL    ADDRES   ;GET ADDRESS OF INST
        JC      G16
        LXI     H,19BAH  ;CONVERT AND STORE
        XCHG
        CALL    CNVRT
        MVI     A,02H
        STA     19B0H
        MVI     A,10H
        STA     19B1H
        MVI     A,03H
        STA     19BEH
        JMP     MAIN
        END
```

```
        STITLE  "BBIMS4 WRITTEN BY JIM MANLEY     FIFTH OF SIX"
        GLOBAL  TRMOUT
        GLOBAL  M40
        GLOBAL  M39
        GLOBAL  CMPDH
        GLOBAL  INITAL
        GLOBAL  MESSAG
        GLOBAL  GETDAT
        GLOBAL  ERROR
        GLOBAL  ADDRES
        GLOBAL  BELL
        GLOBAL  READ
        GLOBAL  ENDIT
        GLOBAL  M15
        GLOBAL  M16
        GLOBAL  M17
        GLOBAL  M18
        GLOBAL  M19
        GLOBAL  M20
        GLOBAL  M21
        GLOBAL  M22
        GLOBAL  M23
        GLOBAL  M24
        GLOBAL  M25
        GLOBAL  M26
        GLOBAL  M27
        GLOBAL  M28
        GLOBAL  M29
INITAL  MVI     A,0FFH
        STA     18EBH
        LXI     H,M22   ;THIS PROG DEFINES ALL CHANNELS FOR TM
                                                         SORTING
        MVI     A,07H   ;DISPLAY AMU LOC MESSAGE
        CALL    GETLOC  ;GETS A BYTE BETWEEN 0 AND 3F
        JC      G2
        STA     18E4H   ;STORE FOR D TO A RECOVERY
G2      MVI     A,19H
        STA     18E5H
        STA     18E7H
        STA     18E9H
        LXI     H,M21   ;DISPLAY DATA LOC MESSAGE
        MVI     A,08H
        CALL    GETLOC  ;GET ANOTHER TM BYTE DEF
        JC      G1
        STA     18E6H   ;STORE FOR DA   STA      18E6H   ;STORE
                                          FOR D TO A RECOVERY
G1      LXI     H,M23   ;DISPLAY SBID LOC MESSAGE
        MVI     A,08H
        CALL    GETLOC  ;GET ANOTHER TM BYTE DEF
        JC      G50
        STA     18E8H   ;STORE FOR SUB ID MATCH
G50     LXI     H,M24   ;DISPLAY DISPLAY/MESSAGE
```

```
        MVI     A,06H
        CALL    MESSAG
        MVI     A,01H
        STA     3806H
        CALL    TRMOUT
        MVI     A,0BFH
        STA     3807H
        CALL    TRMOUT
        LXI     D,1900H ;POINT TO BEGINNING OF DISPLAY/LIST
        CALL    GETALL  ;GET STANDARD DEFINITIONS
        JC      G3      ;JUMP IF NO NEED TO STORE
        SHLD    1908H   ;STORE LOW BYTE LOCATION
        XCHG
        SHLD    190AH   ;STORE HIGH BYTE LOCATION
        LXI     H,M25   ;DISPLAY AMU# MESSAGE
        MVI     A,04H
        CALL    MESSAG
        LXI     H,1809H ;LOAD -HL- WITH STATUS LOC
        CALL    ADDRES  ;GET AMU#
        JC      G3      ;SKIP REST IF CR IS ENTERED
        MOV     A,M     ;DO WE NEED TO SUBTRACT 2000?
        ANI     02H
        JZ      G39     ;JUMP IF WE DONT
        LXI     H,0E000H;SUBTRACT 2000
        DAD     D
        XCHG
G39     BYTE    18H     ;SHIFT -DE- TO THE LEFT 2 TIMES
        BYTE    18H
        MOV     A,E     ;REMOVE 2 LSBITS
        ANI     0FCH
        MOV     E,A
        PUSH    D       ;SAVE AMU#
G37     LXI     H,M26   ;DISPLAY STEP MESSAGE
        MVI     A,04H
        CALL    MESSAG
        LXI     H,1809H
        CALL    GETDAT  ;GET BYTE
        JC      G37     ;JUMP IF ERROR
        POP     H       ;RETRIEVE AMU#
        ANI     03H     ;COMBINE AMU# AND STEP
        ORA     L
        MOV     L,A
        LXI     D,0FFFH ;IS -HL- GREATER THAN MAX
        CALL    CMPDH
        JC      G4      ;JUMP IF TO BIG
        XCHG            ;SHIFT -DE- TO THE LEFT FOUR TIMES
        BYTE    18H
        BYTE    18H
        BYTE    18H
        BYTE    18H
        XCHG
        SHLD    190CH   ;STORE AMU MATCH DATA
```

```
G3      LXI     H,M24     ;DISPLAY DISPLAY 2 MESSAGE
        MVI     A,06H
        CALL    MESSAG
        MVI     A,02H
        STA     3806H
        CALL    TRMOUT
        MVI     A,0BFH
        STA     3807H
        CALL    TRMOUT
        LXI     D,190EH   ;POINT TO BEGINNING OF DISPLAY 2 LIST
        CALL    GETALL    ;GET STANDARD DEFINITIONS
        JC      G33       ;JUMP IF NO NEED TO STORE
        SHLD    1916H     ;STORE LOW BYTE LOCATION
        XCHG
        SHLD    1918H     ;STORE HIGH BYTE LOCATION
G33     LXI     H,M24     ;DISPLAY DISPLAY 3 MESSAGE
        MVI     A,06H     ;THE REST IS THE SAME AS THE DISPLAY 2
                                                       SECTION
        CALL    MESSAG
        MVI     A,03H
        STA     3806H
        CALL    TRMOUT
        MVI     A,0BFH
        STA     3807H
        CALL    TRMOUT
        LXI     D,191AH
        CALL    GETALL
        JC      G34       ;JUMP IF NO NEED TO STORE
        SHLD    1922H
        XCHG
        SHLD    1924H
G34     LXI     H,M24     ;DISPLAY DISPLAY 4 MESSAGE
        MVI     A,06H     ;THE REST IS THE SAME AS THE DISPLAY 2
                                                       SECTION
        CALL    MESSAG
        MVI     A,04H
        STA     3806H
        CALL    TRMOUT
        MVI     A,0BFH
        STA     3807H
        CALL    TRMOUT
        LXI     D,1926H
        CALL    GETALL
        JC      G35
        SHLD    192EH
        XCHG
        SHLD    1930H
G35     XRA     A
        STA     18EBH
        MVI     B,00H     ;THIS BEGINS THE ANALOG DEFINITIONS
G5      INR     B         ;POINT TO NEXT AN CHANNEL
        MOV     A,B
```

```
        CPI     06H     ;IS THIS THE LAST ONE?
        JZ      G10
        PUSH    B
        LXI     H,M28   ;DISPLAY ANALOG MESSAGE
        MVI     A,06H
        CALL    MESSAG
        MVI     A,0BFH
        STA     3807H
        CALL    TRMOUT
        MOV     A,B     ;WHICH ANALOG CHANNEL?
        STA     3806H   ;DISPLAY WHICH ONE
        CALL    TRMOUT
        CPI     01H     ;LOAD -HL- WITH STARTING OF THE
                                          CHANNELS LIST
        JZ       11
        CPI     03H
        JC      G12
        JZ      G13
        CPI     05H
        JC      G14
        LXI     D,1962H ;BEGINNING OF CHANNEL 5
G15     PUSH    H       ;SAVE BEGINNING POINTER
        CALL    GETALL  ;GET STANDARD DEFINITION
        XCHG
        POP     D
        POP     B
        JC      G5      ;JUMP IF NO NEED TO STORE
        MOV     A,B     ;GET CHANNEL NUMBER
        CPI     01H
        JZ      G6
        CPI     03H
        JC      G7
        JZ      G8
        CPI     05H
        JC      G9
        SHLD    196AH   ;STORE BYTE LOCATION
G10     LXI     H,M29   ;DISPLAY TTY LOC MESSAGE
        MVI     A,07H
        CALL    MESSAG
        LXI     H,1809H ;PUT STATUS ADDRESS INTO -HL-
        MVI     A,0FFH
        STA     1849H
        CALL    GETDAT
        JC      G73     ;JUMP IF ERROR IN SUB
        CPI     40H     ;IS BYTE BETWEEN 0 TO 3F?
        JNC     G10     ;JUMP IF ITS NOT
        ADI     6EH     ;ADD OFFSET
        STA     19AEH   ;STORE BYTE LOCATION
        MVI     A,19H
        STA     19AFH
G81     LXI     H,M39
        MVI     A,07H
```

```
        CALL    MESSAG
        CALL    READ
        CPI     9BH
        JZ      G72
        CPI     8DH
        JZ      G74
        CPI     0CEH
        JZ      G82
        CPI     0D9H
        JNZ     G81
        XRA     A
        STA     1829H
G74     CALL    ENDIT
        RET
G82     MVI     A,0FFH
        STA     1829H
        JMP     G74
G73     ORA     A       ;IS -ACC- ZERO
        JZ      G81     ;JUMP IF IT IS AND END
        JMP     G10
GETLOC  CALL    MESSAG  ;THIS SUB GETS A BYTE BETWEEN 0 AND 3F
        LXI     H,1809H ;LOAD STATUS ADDRESS INTO -HL-
        MVI     A,0FFH
        STA     1849H
        CALL    GETDAT
        RC              ;RETURN IF ERROR
        CPI     40H     ;IS IT LESS THAN 40?
        JNC     G40     ;JUMP IF ERROR
        ADI     6EH     ;ADD AN OFFSET
        RET
G40     CALL    ERROR
        STC
        RET
G4      LXI     H,M27   ;DISPLAY BAD AMU MESSAGE
        MVI     A,07H
        CALL    MESSAG  ;RING BELL
        LXI     H,3FFFH ;RING BELL
        CALL    BELL
        RET
G11     LXI     D,1932H
        JMP     G15
G12     LXI     D,193EH
        JMP     G15
G13     LXI     D,194AH
        JMP     G15
G14     LXI     D,1956H
        JMP     G15
G6      SHLD    193AH
        JMP     G36
G7      SHLD    1946H
        JMP     G36
G8      SHLD    1952H
```

```
        JMP     G36
G9      SHLD    195EH
G36     LDAX    D
        ORI     01H
        STAX    D
        JMP     G5
GETALL  CALL    READ     ;THIS SUB GETS STANDARD DEFINITIONS
        CPI     9BH      ;IS ENTRY AN ESC?
        JZ      G72
        CPI     8DH      ;IS ENTRY A CR?
        JZ      G17      ;JUMP IF IT IS AND SKIP ALL DEFINITIONS
        CPI     0CEH     ;IS ENTRY A N?
        JZ      G16      ;JUMP IF IT IS AND REMOVE ALL
                                                   DEFINITIONS
        CPI     0D9H     ;IS ENTRY A Y?
        JNZ     GETALL   ;ERROR IN ENTRY SRART AGAIN
G18     LXI     H,M15    ;DISPLAY DATAVAL? MESSAG
        MVI     A,08H
        CALL    MESSAG
        CALL    READ     ;GET YES OR NO ENTRY
        CPI     9BH      ;IS ENTRY AN ESC
        JZ      G72
        CPI     0CEH     ;IS IT A N?
        JZ      G19      ;JUMP IF IT IS
        CPI     0D9H
        JNZ     G18      ;ERROR IN ENTRY TRY AGAIN
        MVI     A,0C0H   ;LOAD -ACC- WITH FLAGS USED AND DATA
                                                       VALID
G20     STAX    D        ;STORE FLAGS
G21     LXI     H,M16    ;DISPLAY DECIMAL? MESSAGE
        MVI     A,08H
        CALL    MESSAG
        CALL    READ     ;GET YES OR NO ENTRY
        CPI     9BH      ;IS ENTRY AN ESC?
        JZ      G72
        CPI     0CEH     ;IS ENTRY A N?
        JZ      G22      ;JUMP IF IT IS
        CPI     0D9H
        JNZ     G21      ;ERROR IN ENTRY TRY AGAIN
        MVI     B,20H    ;LOAD -B- WITH FLAG DECIMAL
G23     LDAX    D        ;LOAD FLAG INTO FLAG BUFFER
        ORA     B
        STAX    D
G84     LXI     H,M40    ;DISPLAY MATCH? MESSAGE?
        MVI     A,06H
        CALL    MESSAG
        CALL    READ
        CPI     9BH
        JZ      G72
        CPI     0CEH
        JZ      G79
        CPI     0D9H
```

```
        JNZ     G84
        MVI     B,08H
G85     LDAX    D
        ORA     B
        STAX    D
G24     LXI     H,M17   ;DISPLAY SUB IDS MESSAGE
        MVI     A,08H
        CALL    MESSAG
        CALL    READ
        CPI     9BH     ;IS ENTRY AN ESC?
        JZ      G72
        CPI     0CEH    ;IS ENTRY A N?
        JZ      G25     ;JUMP IF IT IS
        CPI     0D9H    ;IS ENTRY A Y?
        JNZ     G24     ;ERROR IN ENTRY TRY AGAIN
        MVI     B,10H   ;LOAD -B- WITH FLAG SUB IDS
        LDAX    D       ;LOAD FLAG INTO FLAG BUFFER
        ORA     B
        STAX    D
        PUSH    D       ;SAVE BUFFER ADDRESS
        MVI     B,00H   ;SUB ID COUNTER
G28     LXI     H,M18   ;DISPLAY SUB ID MESSAGE
        MVI     A,06H
        CALL    MESSAG
        LXI     H,1809H ;LOAD -HL- WITH STATUS ADDRESS
        PUSH    B       ;SAVE SUB ID COUNTER
        PUSH    D       ;SAVE SUB ID STORAGE POINTER
        MVI     A,0FFH  ;TELL SUB GET DALL TO RETURN ON 00
                                                          ENTRY
        STA     1849H
        CALL    GETDAT  ;GET SUB ID NUMBER
        POP     D
        POP     B
        JC      G27     ;JUMP IF NO MORE ENTRIES
        INX     D       ;SET UP TO STORE SUB ID
        INR     B
        STAX    D
        MOV     A,B     ;CAN THERE BE ANY MORE SUB IDS ENTERED
        CPI     07H
        JNZ     G28     ;JUMP IF THERE CAN
G27     POP     H       ;GET BACK FLAG POINTER
        MOV     A,B     ;LOAD SUB IF COUNTER INTO FLAGS
        ORA     M
        MOV     M,A
G26     LDA     18EBH
        ORA     A
        JZ      G83
        LXI     H,M19   ;DISPLAY LOW BYTE LOC MESSAGE
        MVI     A,08H
        CALL    MESSAG
        LXI     H,1809H ;LOAD -HL- WITH STATUS ADDRESS
G29     MVI     A,0FFH  ;TELL SUB GETDAT TO RETURN
```

```
        STA     1849H
        CALL    GETDAT
        CPI     40H     ;IS IT A GOOD ENTRY?
        JNC     G30     ;JUMP IF NOT
        ADI     6EH     ;ADD AN OFFSET
        MOV     E,A     ;SAVE LOW BYTE ADDRESS
        MVI     D,19H
G83     PUSH    D
        LXI     H,M20   ;DISPLAY MSBYTE LOCATION MESSAGE
        MVI     A,08H
        CALL    MESSAG
        LXI     H,1809H ;LOAD -HL- WITH STATUS ADDRESS
G31     MVI     A,0FFH  ;TELL SUB GETDAT TO RETURN UPON 00
        STA     1849H
        CALL    GETDAT
        CPI     40H     ;ENTRY MUST BE LESS THAN 40H
        JNC     G32     ;JUMP IF ITS NOT
        ADI     6EH     ;ADD AN OFFSET
        MOV     E,A     ;PUT HIGH BYTE INTO -DE-
        MVI     D,19H
        POP     H       ;PUT LOW BYTE INTO -HL-
        RET
G79     MVI     B,00H
        JMP     G85
G72     CALL    ENDIT
        LHLD    182AH   ;GET HOME ADDRESS AND RET
        PCHL
G32     CALL    ERROR
        JMP     G31
G30     CALL    ERROR
        JMP     G29
G25     MVI     B,00H   ;RESET SUBID FLAG
        LDAX    D
        ORA     B
        STAX    D
        JMP     G26     ;GET BYTE LOCATIONS
G22     MVI     B,00H   ;RETSET DECIMAL FLAG
        JMP     G23
G19     MVI     A,80H   ;RESET DATA VALID AND SET USED FLAGS
        JMP     G20
G16     XRA     A       ;CLEAR ALL FLAGS
        STAX    D
G17     STC             ;UPON RETURN DONT STORE BYTE LOCATIONS
        RET
        END
```

```
        STITLE  "BBIMS5 WRITTEN BY JIM MANLEY     SIXTH OF SIX"
        GLOBAL  CLEAR
        GLOBAL  G48
        GLOBAL  CMPDH
        GLOBAL  LDBUFF
        GLOBAL  TTYLNK
        GLOBAL  BINDEC
        GLOBAL  MAIN
        GLOBAL  MOVE
        GLOBAL  FRAME
ALLDEF  PUSH    D        ;GET FLAGS USED DECIMAL SUBID AND DATA
                                           WORD UPON RETURNING
        PUSH    B
        MOV     A,M      ;UPON ENTERING -HL- CONTAINS POINTER TO
                                                   TOP OF LIST
        RLC              ;CHECK IS DISPLAY IS USED
        CMC
        JC      G47      ;JUMP IF ITS NOT
        RLC              ;CHECK IF DATA VALID NEEDED
        JC      G41      ;JUMP IF IT IS NEEDED
G42     RLC              ;CHECK FOR SUBIDS NEEDED
        RLC
        JC      G43
G40     RLC
        JNC     G46
        LDA     1830H
        RLC
        CMC
        JC      G47
G46     LXI     D,0008H ;ADD 8 TO -HL- TO GET TO DATA LOCATIONS
        DAD     D
        MOV     E,M      ;GET ADDRESS OF FIRST BYTE
        INX     H
        MOV     D,M
        LDAX    D        ;GET LOW BYTE
        MOV     C,A
        INX     H        ;GET ADDRESS OF LAST BYTE
        MOV     E,M
        INX     H
        MOV     D,M
        LDAX    D        ;GET HIGH BYTE
        MOV     H,A      ;PUT WORD INTO -HL-
        MOV     L,C
        ORA     A        ;CLEAR CY BIT
G47     POP     B
        POP     D
        RET
LDBUFF  LDA     18E3H    ;CHECK FOR A BUFFER FULL
        ORA     A
        JZ      LDBUFF
        DI
        LDA     18E3H    ;DONT LET BUFFER FLAGS CHANGE
```

```
        RAL                ;IS BUFFER 1 FULL?
        JC      G51        ;JUMP IF IT IS
        LXI     H,18A0H    ;LOAD -HL-DE-BC-- WITH MOVE PARAMETERS
        LXI     D,18DFH
G52     XRA     A
        STA     18E3H      ;RESET BUFFER FULL FLAGS
        EI
        LXI     B,196EH
        CALL    MOVE
        LXI     H,196EH    ;CHECK LAST WORK IN TM FOR SYNC
        LDA     18E2H      ;GET NUMBER OF BYTES IN FRAME
        DCR     A
        ADD     L          ;ADD NUMBER OF BYTES TO POINTER
        MOV     L,A
        RET
MAIN    CALL    CLEAR      ;CLEAR ALL DISPLAYS
        DI                 ;START OF MAIN TM OPERATING SYSTEM
        XRA     A          ;FIRST FRAME THE PCM TRAIN
        STA     18E3H
        LXI     H,1860H     ;STORE BEGINNING OF TM BUFFER IN
                                                        POINTER
        SHLD    18E0H
        CALL    FRAME
G48     MVI     B,00H      ;LET ONE ERROR IN FRAMEING GO BY
G49     PUSH    B
        CALL    LDBUFF     ;LOAD THE PCM BUFFER
        MOV     A,M        ;GET LAST BYTE
        CPI     90H        ;CHECK FOR SYNC
        JNZ     G53        ;JUMP IF NOT RIGHT
        DCX     H
        MOV     A,M        ;GET SECOND TO LAST BYTE
        CPI     0EBH
        JNZ     G53        ;JUMP IF NO SYNC
        MVI     A,04H      ;TURN ON SYNC LED
        OUT     0B3H
        POP     B          ;CORRECT THE STACK
        LDA     196EH      ;CHECK FOR DATA VALID
        RLC
        JNC     G55        ;JUMP IF NO VALID DATA
        LHLD    18E4H
        MOV     D,M
        INX     H
        MOV     E,M
        XCHG
        BYTE    10H,10H,10H,10H,10H,10H
        MOV     A,L
        OUT     0F1H
        BYTE    10H,10H
        MOV     A,L
        OUT     0F0H
        LHLD    18E6H
        MOV     D,M
```

```
        INX     H
        MOV     E,M
        XCHG
        MOV     A,H
        RLC
        JC      G56
        MVI     A,09H
G57     OUT     0B3H
        MOV     A,L
        OUT     0F4H
        BYTE    10H,10H,10H,10H
        MOV     A,L
        OUT     0F3H
        BYTE    10H,10H,10H,10H
        MOV     A,L
        ANI     07FH
        OUT     0F2H
        LDA     196EH    ;OUTPUT DATA VALID CPU DWN LEDS
        CMA
        ANI     0E0H
        MOV     B,A
        IN      0B2H
        ANI     1FH
        ORA     B
        OUT     0B2H
G55     LDA     196EH    ;CHECK FOR BURST READY FLAG
        ANI     08H
        JZ      G80      ;JUMP AND TURN OFF BELL
        MVI     A,06H    ;TURN ON BELL IF GETTING READY TO DUMP
G78     OUT     0B3H
        LDA     196EH    ;CHECK DATA VALID FLAGS
        RAL
        CMC
        MVI     A,07H
        RAL              ;SET UP TO TURN ON OR OFF DATA VALID
                                                          LED
        OUT     0B3H
        LDA     196EH
        ANI     04H      ;CHECK FOR TTY ACTIVE
        JNZ     TTYLNK   ;JUMP IF ACTIVE
        XRA     A        ;CLEAR MATCH FLAG
        STA     1830H
        LDA     196EH
        RLC
        JNC     G50
        LHLD    190CH
        CALL    CMPDH
        MVI     A,00H
        JNZ     G87
        CMA
G87     STA     1830H
G50     LXI     H,1900H ;LOAD -HL- WITH START OF DISPLAY 1
```

PARAMETERS

```
        CALL    ALLDEF   ;GET DATA
        JC      G60      ;JUMP IF NO UPDATE NEEDED
        LDA     1900H    ;CHECK FOR MATCH FLAG
        ANI     08H
        JZ      G59      ;JUMP IF NO MATCH
        BYTE    10H,10H,10H,10H,10H,10H
        MOV     A,H
        ANI     03H
        MOV     H,A
G59     LDA     1900H    ;CHECK FOR DECIMAL CONVERSION
        ANI     20H
        XCHG
        CNZ     BINDEC   ;CALL IF CONVERSION NEEDED
        MOV     A,D      ;NOW DISPLAY ON DISPLAY 1
        ANI     0F0H
        RRC
        RRC
        RRC
        RRC
        STA     181BH
        MOV     A,D
        ANI     0FH
        STA     181AH
        MOV     A,E
        ANI     0F0H
        RRC
        RRC
        RRC
        RRC
        STA     1819H
        MOV     A,E
        ANI     0FH
        STA     1818H
G60     LXI     H,190EH  ;LOAD -HL- WITH POINTER FOR DISPLAY 2
        CALL    ALLDEF
        JC      G62      ;JUMP IF BY PASSED
        LDA     190EH
        ANI     08H
        JZ      G88
        MOV     A,H
        RAL
        JNC     G89
        CMC
        RAR
        MOV     H,A
        MVI     A,10H
G76     STA     1827H
        LXI     D,9999
        CALL    CMPDH
        JNC     G90
        XRA     A
```

```
G91     OUT     0B3H
G88     XCHG
        LDA     190EH    ;IS DECIMAL CONVERSION NEEDED?
        ANI     20H
        CNZ     BINDEC   ;CALL IF CONVERSION NEEDED
        MOV     A,D      ;DISPLAY ON DISPLAY 2
        ANI     0F0H
        RRC
        RRC
        RRC
        RRC
        STA     1826H
        MOV     A,D
        ANI     0FH
        STA     1825H
        MOV     A,E
        ANI     0F0H
        RRC
        RRC
        RRC
        RRC
        STA     1824H
        MOV     A,E
        ANI     0FH
        STA     1823H
G62     LXI     H,191AH  ;LOAD -HL- WITH POINTER FOR DISPLAY 3
        CALL    ALLDEF
        JC      G61      ;JUMP IF BYPASSED
        XCHG
        LDA     191AH    ;IS DECIMAL CONVERSION NEEDED
        ANI     20H
        CNZ     BINDEC   ;CALL IF IT IS
        MOV     A,D      ;DISPLAY ON DISPLAY 3
        ANI     0F0H
        RRC
        RRC
        RRC
        RRC
        STA     1821H
        MOV     A,D
        ANI     0FH
        STA     1820H
        MOV     A,E
        ANI     0F0H
        RRC
        RRC
        RRC
        RRC
        STA     181FH
        MOV     A,E
        ANI     0FH
        STA     181EH
```

```
G61     LXI     H,1926H ;LOAD -HL- WITH POINTER FOR DISPLAY 4
        CALL    ALLDEF
        JC      G67     ;JUMP IF BY PASSED
        XCHG
        LDA     1926H   ;DECIMAL CONVERSION NEEDED?
        ANI     20H
        CNZ     BINDEC  ;CALL IF IT IS
        MOV     A,D     ;DISPLAY ON DISPLAY 4
        ANI     0F0H
        RRC
        RRC
        RRC
        RRC
        STA     3800H
        MOV     A,D
        ANI     0FH
        STA     3801H
        MOV     A,E
        ANI     0F0H
        RRC
        RRC
        RRC
        RRC
        STA     3802H
        MOV     A,E
        ANI     0FH
        STA     3803H
G67     RST     1       ;UPDATE HEX DISPLAYS
        LXI     H,1932H ;LOAD -HL- WITH POINTER FOR ANALOG 1
        CALL    ALLDEF
        JC      G68     ;JUMP IF BY PASSED
        MOV     A,L
        OUT     0F5H    ;OUTPUT TO A1
G68     LXI     H,193EH ;LOAD -HL- WITH ANALOG 2 POINTER
        CALL    ALLDEF
        JC      G69     ;JUMP IF BYPASSED
        MOV     A,L
        OUT     0F6H    ;OUTPUT TO AN2
G69     LXI     H,194AH ;LOAD -HL- WITH ANALOG 3 POINTER
        CALL    ALLDEF
        JC      G70     ;JUMP IF BYPASSED
        MOV     A,L
        OUT     0F7H    ;OUTPUT TO AN3
G70     LXI     H,1956H ;LOAD -HL- WITH ANALOG 4 POINTER
        CALL    ALLDEF
        JC      G71     ;JUMP IF BYPASSED
        MOV     A,L
        OUT     0F8H    ;OUTPUT TO ANY
G71     LXI     H,1962H ;LOAD -HL- WITH ANALOG 5 POINTER
        CALL    ALLDEF
        JC      G48     ;JUMP IF BYPASSED
        MOV     A,L
```

```
        OUT     0F9H     ;OUTPUT TO AN5
        JMP     G48
G80     MVI     A,07H    ;TURN BELL OFF
        JMP     G78
G41     MOV     B,A      ;CHECK FOR DATA VALID
        LDA     196EH
        RLC
        MOV     A,B
        CMC
        JC      G47      ;JUMP IF IT ISNT
        JMP     G42
G43     PUSH    H        ;SAVE FIRST POINTER LOCATION
        PUSH    PSW
        MOV     A,M      ;GET NUMBER OF SUBIDS IN LIST
        ANI     07H
        MOV     B,A      ;STORE IN -B-
        XCHG
        INX     D        ;POINT TO FIRST SUBID
        LHLD    18E8H    ;GET LOCATION OF PRESENT SUBID
        MOV     A,M      ;GET PRESENT SUBID
        XCHG
G44     CMP     M        ;COMPARE PRESENT SUBID TO SUBID LIST
        JZ      G45      ;JUMP IF THERE IS A MATCH
        DCR     B        ;DECREAMENT NUMBER OF SUBIDS IN THE
                                                          LIST
        JZ      G77      ;JUMP IF NO MORE
        INX     H        ;POINT TO NEXT SUBID
        JMP     G44
G89     MVI     A,0A0H
        JMP     G76
G90     MVI     A,01H
        JMP     G91
G77     POP     PSW
        POP     H
        STC
        JMP     G47
G45     POP     PSW      ;GET BACK FIRST POINTER LOCATION
        POP     H
        JMP     G40
G51     LXI     H,1860H  ;SET UP TO MOVE BUFFER 1 TO ANOTHER
                                                        BUFFER
        LXI     D,189FH
        JMP     G52
G53     POP     B        ;GET BACK -B-
        MVI     A,05H    ;TURN OFF SYNC LED
        OUT     0B3H
        MOV     A,B
        MVI     B,0FFH   ;SET SECOND TIME THROUGH FLAG
        RLC              ;IS THIS THE SECOND TIME THROUGH LOOP?
        JC      MAIN
        JMP     G49
G56     MVI     A,08H    ;TURN ON NEG LED
```

```
JMP     G57
END
```

```
SECTION TABLES
STITLE "LOOKUP TABLES AND MESSAGES FOR BBIMS BY J
                                                   MANLEY"
GLOBAL  M40
GLOBAL  M41
GLOBAL  M42
GLOBAL  M43
GLOBAL  M44
GLOBAL  M45
GLOBAL  M39
GLOBAL  LOOKP
GLOBAL  M30
GLOBAL  M31
GLOBAL  M32
GLOBAL  M33
GLOBAL  M34
GLOBAL  M35
GLOBAL  M36
GLOBAL  M37
GLOBAL  M38
GLOBAL  MODE
GLOBAL  M12A
GLOBAL  BIASP
GLOBAL  BIASS
GLOBAL  MASK
GLOBAL  IDNUM
GLOBAL  RPAGE
GLOBAL  RBOOK
GLOBAL  NPAGE
GLOBAL  DUMP
GLOBAL  WAIT
GLOBAL  GOTOBB
GLOBAL  CONT
GLOBAL  MAIN
GLOBAL  INITAL
GLOBAL  RATIO
GLOBAL  LOP
GLOBAL  TIME
GLOBAL  AMU
GLOBAL  NUMLK
GLOBAL  BIN
GLOBAL  M1
GLOBAL  M2
GLOBAL  M3
GLOBAL  M4
GLOBAL  M5
GLOBAL  M6
GLOBAL  M7
GLOBAL  M8
GLOBAL  M9
GLOBAL  M10
GLOBAL  M11
```

```
        GLOBAL  M12
        GLOBAL  M13
        GLOBAL  M14
        GLOBAL  M15
        GLOBAL  M16
        GLOBAL  M17
        GLOBAL  M18
        GLOBAL  M19
        GLOBAL  M20
        GLOBAL  M21
        GLOBAL  M22
        GLOBAL  M23
        GLOBAL  M24
        GLOBAL  M25
        GLOBAL  M26
        GLOBAL  M27
        GLOBAL  M28
        GLOBAL  M29
        GLOBAL  FEPROM
        GLOBAL  SWITCH
        GLOBAL  BAUD
        GLOBAL  GOTO
        GLOBAL  BNRY
        GLOBAL  DCML
        GLOBAL  DIRECT
        GLOBAL  INDRCT
        GLOBAL  MOVEM
        GLOBAL  FILLM
        GLOBAL  DISPL
        GLOBAL  ALTR
        GLOBAL  COMPA
        GLOBAL  COMPD
LOOKP   ASCII   "BINARY "
        WORD    BNRY
        BYTE    0FFH,0FFH
        BYTE    92H,0F4H
        ASCII   "DECIMAL "
        WORD    DCML
        BYTE    0FFH,0FFH
        BYTE    93H,0F3H
        ASCII   "DRCT "
        WORD    DIRECT
        BYTE    0FFH,0FFH
        BYTE    94H,0F6H
        ASCII   "INDRCT "
        WORD    INDRCT
        BYTE    0FFH,0FFH
        BYTE    95H,0F4H
        ASCII   "MOVE "
        WORD    MOVEM
        BYTE    03H,18H
        BYTE    96H,0F6H
```

```
ASCII   "FILL "
WORD    FILLM
BYTE    02H,18H
BYTE    97H,0F6H
ASCII   "DISPLAY "
WORD    DISPL
BYTE    00H,18H
BYTE    0FFH,0FFH
ASCII   "ALTER "
WORD    ALTR
BYTE    01H,18H
BYTE    98H,0F5H
ASCII   "COMPA "
WORD    COMPA
BYTE    05H,18H
BYTE    99H,0F5H
ASCII   "COMPD "
WORD    COMPD
BYTE    05H,18H
BYTE    9AH,0F5H
ASCII   "GO "
WORD    GOTO
BYTE    07H,18H
BYTE    91H,0F8H
ASCII   "BAUD "
WORD    BAUD
BYTE    4FH,18H
BYTE    9CH,0F6H
ASCII   "FEPROM "
WORD    FEPROM
BYTE    07H,18H
BYTE    9DH,0F4H
ASCII   "SWITCH "
WORD    SWITCH
BYTE    4FH,18H
BYTE    9EH,0F4H
ASCII   "MAIN "
WORD    MAIN
BYTE    09H,18H
BYTE    9FH,0F6H
ASCII   "INITIAL "
WORD    INITAL
BYTE    09H,18H
BYTE    0A0H,0F3H
ASCII   "RPAGE "
WORD    RPAGE
BYTE    09H,18H
BYTE    0B3H,0F5H
ASCII   "RBOOK "
WORD    RBOOK
BYTE    09H,18H
BYTE    0B4H,0F5H
```

```
ASCII   "NPAGE "
WORD    NPAGE
BYTE    09H,18H
BYTE    0ACH,0F5H
ASCII   "DUMP "
WORD    DUMP
BYTE    09H,18H
BYTE    0ADH,0F6H
ASCII   "WAIT "
WORD    WAIT
BYTE    09H,18H
BYTE    0AEH,0F6H
ASCII   "CONT "
WORD    CONT
BYTE    09H,18H
BYTE    0AFH,0F6H
ASCII   "GOBB "
WORD    GOTOBB
BYTE    09H,18H
BYTE    0ABH,0F6H
ASCII   "MASK "
WORD    MASK
BYTE    09H,18H
BYTE    0A1H,0F6H
ASCII   "IDNUM "
WORD    IDNUM
BYTE    09H,18H
BYTE    0A2H,0F5H
ASCII   "BIASP "
WORD    BIASP
BYTE    09H,18H
BYTE    0A3H,0F5H
ASCII   "BIASS "
WORD    BIASS
BYTE    09H,18H
BYTE    0A9H,0F5H
ASCII   "MODE "
WORD    MODE
BYTE    09H,18H
BYTE    0A4H,0F6H
ASCII   "RATIO "
WORD    RATIO
BYTE    09H,18H
BYTE    0A5H,0F5H
ASCII   "LOP "
WORD    LOP
BYTE    09H,18H
BYTE    0A6H,0F7H
ASCII   "TIME "
WORD    TIME
BYTE    09H,18H
BYTE    0A7H,0F6H
```

```
        ASCII   "AMU "
        WORD    AMU
        BYTE    09H,18H
        BYTE    0A8H,0F7H
        WORD    0FFFFH
M1      ASCII   "OLD MIN"
M2      ASCII   "OLD MAX"
M3      ASCII   "NEW MIN"
M4      ASCII   "MIN>MAX"
M5      ASCII   "MIN ADDRESS"
M6      ASCII   "MAX ADDRESS"
M7      ASCII   "BAD DATA?"
M8      ASCII   "MODE?"
M9      ASCII   "ERROR"
M10     ASCII   "END"
M11     ASCII   "AGAINST"
BIN     WORD
            1,2,4,8,16H,32H,64H,128H,256H,512H,1024H,2048H,4096H
        WORD    8192H,6384H,2768H,5536H
NUMLK   BYTE    03H       ;HEX SEGMENT LOOKUP TABLE     0
        BYTE    9FH       ;1
        BYTE    25H       ;2
        BYTE    0DH       ;3
        BYTE    99H       ;4
        BYTE    49H       ;5
        BYTE    41H       ;6
        BYTE    1FH       ;7
        BYTE    01H       ;8
        BYTE    19H       ;9
        BYTE    11H       ;A
        BYTE    0C1H      ;B
        BYTE    63H       ;C
        BYTE    85H       ;D
        BYTE    61H       ;E
        BYTE    71H       ;F
        BYTE    0FDH      ;-
M12     ASCII   "BAD "
M12A    ASCII   "ADDRESS"
M13     ASCII   "RATE"
M14     ASCII   "BJC-2"
M15     ASCII   "DATAVAL?"
M16     ASCII   "DECIMAL?"
M17     ASCII   "SUB IDS?"
M18     ASCII   "SUB ID#"
M19     ASCII   "LBYT LOC"
M20     ASCII   "HBYT LOC"
M21     ASCII   "DATA LOC"
M22     ASCII   "AMU LOC"
M23     ASCII   "SBID LOC"
M24     ASCII   "DSPLAY"
M25     ASCII   "AMJ #"
M26     ASCII   "STEP"
```

```
M27     ASCII   "BAD AMU"
M28     ASCII   "ANALOG"
M29     ASCII   "TTY LOC"
M30     ASCII   "START?"
M31     ASCII   "NUMBER"
M32     ASCII   "ENDING?"
M33     ASCII   "DOWN?"
M34     ASCII   "AMU SWP?"
M35     ASCII   "TOTIONS?"
M36     ASCII   "ACCUM?"
M37     ASCII   "SWITCH?"
M38     ASCII   "BIAS"
M39     ASCII   "PCMLNK?"
M40     ASCII   "MATCH?"
M41     ASCII   "STEPING?"
M42     ASCII   "VALUE"
M45     ASCII   "RPETOIRE"
M43     ASCII   "PROGRAM"
M44     ASCII   "INSTRCT"
        END
```

V. REFERENCES

1. Rochefort, J.S. and Sukys, R., "Electronics for Rocket Borne Quadrupole Cluster Ion Mass Filter", Final Report, Contract No. F19628-76-C-0256, October 1978, AFGL-TR-78-0292.

2. Gerousis, V.S., "A Programmable Control Unit for a Balloon-Borne Mass Spectrometer Based on Intel 8085A Microprocessor", Scientific Report No. 1, Contract No. AF19628-78-C-0218, September 1979, AFGL-TR-79-0225.

3. Palasek, T., "An RF Oscillator for Rocket-Borne and Balloon-Borne Mass Spectrometers", Scientific Report No. 2, Contract No. AF19628-78-C-0218, September 1979, AFGL-TR-79-0226.

4. Sukys, R. and Rochefort, J.S., "Control and Data Transmissions System for a Balloon-Borne Ion Mass Spectrometer", Proceedings International Telemetering Conference, October 1980, Vol. XVI, pp. 335-341.

5. Sukys, R. and Rochefort, J.S. "Control and Data Transmission System for a Balloon-Borne Ion Mass Spectrometer", Scientific Report No. 3, Contract No. AF19628-78-C-0218, October 1980, AFGL-TR-81-0202.

VI. PERSONNEL

A. A list of the engineers and student assistants who contributed to the work reported is given below:

J. Spencer Rochefort, Professor of Electrical Engineering, Department Chairman, Principal Investigator.

Raimundas Sukys, Senior Research Associate, Engineer.

Thomas Wheeler, Research Assistant, Engineer.

James R. Manley, Jr., Project Assistant.

B. ACKNOWLEDGEMENT

The authors would like to give special recognition to the contributions of Project Assistant James Manley. Although a student, he brought to the project such advanced and specialized knowledge in the area of computer software and instrumentation that most of the software and CPU design were his contribution.

VII. RELATED CONTRACTS AND PUBLICATIONS

F19628-74-C-0042	1 September 1973 through 31 July 1976
F19628-76-C-0256	1 August 1976 through 31 October 1978
F19628-78-C-0218	15 September 1978 through 14 September 1981
F19628-81-C-0162	15 September 1981 through present.

Sukys, R. and Goldberg, S., "Control Circuits for a Rocket Payload Neutralization and Other Topics", Scientific Report No. 1, Contract No. F19628-74-C-0042, October 1974, AFGL-TR-74-0580.

Sukys, R., Rochefort, J.S. and Goldberg, S., "Bias and Signal Processing Circuits for a Mass Spectrometer in the Project EXCEDE: SWIR Experiment", Scientific Report No. 2, October 1975, Contract No. F19628-74-C-0042, AFGL-TR-76-0060.

Rochefort, J.S. and Sukys, R., "Instrumentation Systems for Mass Spectrometers", Final Report, September 1976, Contract No. F19628-74-C-0042, AFGL-TR-76-0200.

Rochefort, J.S. and Sukys, R., "A Digital Control Unit for a Rocket Borne Quadrupole Mass Spectrometer", Scientific Report No. 1, April 1978, Contract No. F19628-76-C-0256, AFGL-TR-78-0106.

Rochefort, J.S. and Sukys, R., "Electronics for Rocket Borne Quadrupole Cluster Ion Mass Filter", Final Report, Contract No. F19628-76-C-0256, October 1978, AFGL-TR-78-0292.

Gerousis, V.S., "A Programmable Control Unit for a Balloon-Borne Mass Spectrometer Based on Intel 8085A Microprocessor", Scientific Report No. 1, Contract No. AF19628-78-C-0218, September 1979, AFGL-TR-79-0225.

Palasek, T., "An RF Oscillator for Rocket-Borne and Balloon-Borne Mass Spectrometers", Scientific Report No. 2, Contract No. AF19628-78-C-0218, September 1979, AFGL-TR-79-0226.

Sukys, R. and Rochefort, J.S., "Control and Data Transmissions System for a Balloon-Borne Ion Mass Spectrometer", Proceedings International Telemetering Conference, October 1980, Vol. XVI, pp. 335-341.

Sukys, R. and Rochefort, J.S. "Control and Data Transmission System for a Balloon-Borne Ion Mass Spectrometer", Scientific Report No. 3, Contract No. AF19628-78-C-0218, October 1980, AFGL-TR-81-0202.